Statistical Computations Based Upon Algebraically Specified Models

J. E. Schlater
United States Steel Corporation, Monroeville, Pa.*
AND
W. J. Hemmerle
University of Rhode Island, Kingston, R.I.

Based upon a machine-readable statistical model and related symbolic specifications, an efficient method of performing calculations for statistical models of a balanced complete nature is presented. Fixed, mixed, and random analysis of variance models are considered. A procedure for obtaining variance components and calculated F statistics for the model terms is included.

Introduction

Any practical statistical computing system must employ special techniques to handle computations for balanced comp'ete experimental structures. Although the general theory of the linear hypothesis applies to models with fixed effects, computer storage and time considerations make a regression approach very inefficient. Furthermore, the frequency of analyses associated with such structures justifies considerable attention to this case.

In this paper, computational methods are presented for accepting as input an algebraic statistical model along with related symbolic specifications and for performing the statistical calculations dictated by the model. A comprehensive treatment is given for analysis of variance of balance complete structures of fixed, mixed, and random models. It is an extension of a previous article [1] that discussed a notational scheme and related algorithm for fixed models. However, in [1] certain restrictions are imposed in writing the algebraic model, and in some cases pooling is required to obtain an appropriate sum of squares. The methods given in the present paper essentially remove these alphabetic restrictions and obviate any need for pooling sums of squares. These methods have also been extended to handle covariance models as discussed in [4].

A time comparison of algorithms for computing sums of squares was made between the method of factorial decomposition given in [1] and the general method of computation being presented here. The results of a representative set of problems indicate that the general method of solution is no less than twice as fast as the method of factorial decomposition. Moreover, this ratio increases with the order of the model.

Research leading to this paper was one segment of a study in statistically oriented computer languages and systems supported in part by the National Science Foundation.

* Applied Research Laboratory

Specification of the Model

The algebraic model representing the experimental structure involves effect or factor symbols and subscript symbols with an error term also subscripted. The analysis of variance model on the variate y_{ijkl} where factors P and A are crossed and factor T is nested within P can be written in a form suitable for computer input as

$$Y(IJKL) = P(I) + T(IJ) + A(K) + PA(IK) + TA(IJK) + E(IJKL)$$

The limit of each subscript or number of levels of each factor must also be specified along with a designation of which effects are random. If, for example, the limits for I, J, K and L were 3, 4, 2 and 2 respectively, the factor P were fixed, and the factors T and A were random, then the specifications

$$LIMITS, I = 3, \quad J = 4, \quad K = 2, \quad L = 2$$
$$RANDOM, T, A$$

complete the model definition.

Model specifications discussed in this paper permit any letter to denote effect and subscript symbols. The only rules for writing model terms are those related statistically to balanced complete structures. The algebraic model fully describes the nature of the statistical problem and from it the computations which need to be performed can readily be determined.

Computation Required for Fixed, Mixed and Random Models

Because of the very general method employed, the means, residuals,[1] degrees of freedom and sum of squares are obtainable for all terms of any model representing a balanced complete structure. The method used to compute the sum of squares for a given model term takes a linear combination of the means of the observations to form the residuals for the model term. Experience has dictated the fact that investigators are invariably interested in obtaining the classification means designated by the model. Hence the computation of these means should be considered as an intermediate step in the processing.

Scheffé [3] presents rules for determining the residuals and degrees of freedom which correspond to each line in the analysis of variance table for a balanced complete model. A representation of the residuals and degrees of freedom of the terms of the model described in the previous section is given in Table I.

1. Assignment of Numerical Values

To facilitate the computational procedure, the alphabetic effect and subscript symbols employed in the model statement are assigned numerical values. This permits the

[1] The residuals of a model term are defined as the set of values which, when squared and then summed over all subscripts, yields the sum of squares of the model term.

TABLE I. Relationship between Model Terms, Degrees of Freedom and Residuals

Model term	*Degrees of Freedom*	*Residuals*
$P(I)$	$I - 1$	$Y_{i...} - Y_{....}$
$T(IJ)$	$I(J - 1)$	$Y_{ij..} - Y_{i...}$
$A(K)$	$K - 1$	$Y_{..k.} - Y_{....}$
$PA(IK)$	$(I - 1)(K - 1)$	$Y_{i.k.} - Y_{i...} - Y_{..k.} + Y_{....}$
$TA(IJK)$	$I(J - 1)(K - 1)$	$Y_{ijk.} - Y_{ij..} - Y_{i.k} + Y_{i...}$

TABLE II. Order and Length of Mean Arrays in Core Storage
(Illustrated for four subscripts)

Stage	*Means*	*LSTFI*
Input	Y_{ijkl}	$IJKL$
1	$Y_{ijk.}$	IJK
2	$Y_{ij.l}$	IJL
	$Y_{ij..}$	IJ
3	$Y_{i.kl}$	IKL
	$Y_{i.k.}$	IK
	$Y_{i..l}$	IL
	$Y_{i...}$	I
4	$Y_{.jkl}$	JKL
	$Y_{.jk.}$	JK
	$Y_{.j.l}$	JL
	$Y_{.j..}$	J
	$Y_{..kl}$	KL
	$Y_{..k.}$	K
	$Y_{...l}$	L
	$Y_{....}$	1

algorithm to perform algebraic operations upon the effect and subscript symbols of a given model term. The value 1 is assigned to the last subscript and its corresponding effect symbol if it has one. Assignment of numerical values progresses by powers of 2 such that the first effect and subscript symbols are given the value 2^{n-1} where n is the order of the model. For the model given, the assignment would be

LEFCT	*LSUB*	*LNVES*
P	I	2^3
T	J	2^2
A	K	2
	L	1

2. Computation of Means

A task initial to the analysis of any model by the algorithm is the construction of all possible arrays of means from the data. These means are obtained such that means computed at any given stage are formed by summing the data and all means formed at previous stages over the subscript associated with that stage. These arrays of means follow the data in the same one-dimensional array (FIA) in core storage. The order and size of the arrays are illustrated in Table II.

The assignment of numerical values to the effect and subscript symbols provides a method of determining the location in FIA of any given array of means. There are 2^n arrays in FIA including the data and the over-all mean. The value obtained by the subtraction of the sum of the numerical values of the subscripts occurring in a given array of means from 2^n specifies the relative position in FIA of that array.

3. Inspection of Model Terms

Since the inspection of alphabetic symbols is cumbersome and unnecessary, the model terms are converted to and stored in binary notation. Consider allotting a character of a word in core storage for each possible effect and subscript of each model term. Proceeding from left to right an effect or subscript is denoted by a "1" if it occurs in the model term and a "0" if it does not. The arrays $LMEFT$ and $LMSUB$ denote the effects and subscripts, respectively, of the model terms. These tables take on the form below for the model given allowing for ten factors.

Source	*LMEFT*	*LMSUB*
$P(I)$	1000000000	1000000000
$T(IJ)$	0100000000	1100000000
$A(K)$	0010000000	0010000000
$PA(IK)$	1010000000	1010000000
$TA(IJK)$	0110000000	1110000000

From these tables one can readily determine for a given model term the floating and associated subscripts and their numerical values. In the example, the residuals for the model term $TA(IJK)$ are

$$y_{ijk.} - y_{ij..} - y_{i.k.} + y_{i...}$$

By properly summing the numerical values of the subscripts, a list (call it $LLOCA$) which gives the relative locations in FIA of these four sets of means can be constructed. The numerical values of the floating subscripts are included in each sum and a combinatorial breakdown is performed on the associated subscripts, such that for $TA(IJK)$ the desired values are

LLOCA
8 + 4 + 2..... 14
8 + 4..... 12
8 + 2..... 10
8..... 8

4. Formation of Residuals

The matter of locating means in core storage has been discussed. Once the required arrays are located for a given term, the problem of taking the correct linear combination of the means remains. Clearly a relationship must be developed between the array of means which contains all of the subscripts included in the term—the primary array—and the remaining arrays of means—secondary arrays—of which the residuals for the term are composed. Fortunately, the residuals may be formed in the area in which the data is originally stored. This follows from the fact that for balanced complete structures, *the data enters into the computations of residuals for no more than one model term.* If such a term is given priority in the order of the computations, storage requirements are reduced since additional

array storage is then not required for residual computation. Notice that arrays of means can enter into the computations of residuals for more than one model term.

At the end of the operation of combining means to form the residuals for $TA(IJK)$, the first 24 locations of the FIA array will contain

$$\begin{aligned} FIA(1) &\quad --- \quad y_{111.} - y_{11..} - y_{1.1.} + y_{1...} \\ FIA(2) &\quad --- \quad y_{112.} - y_{11..} - y_{1.2.} + y_{1...} \\ FIA(3) &\quad --- \quad y_{121.} - y_{12..} - y_{1.1.} + y_{1...} \\ &\quad \vdots \\ FIA(24) &\quad --- \quad y_{342.} - y_{34..} - y_{3.2.} + y_{3...} \end{aligned}$$

These linear combination of means are formed sequentially as indicated below

$y_{ijk.}$ (initialization)
$y_{ijk.} - y_{ij..}$
$y_{ijk.} - y_{ij..} - y_{i.k.}$
$y_{ijk.} - y_{ij..} - y_{i.k.} + y_{i...}$

Consider an associated pair of mappings—a primary map and a secondary map. The primary mapping gives the location of a certain mean of the primary array stored in one-dimensional array form. The mapping for the primary array $y_{ijk.}$ can be expressed as:

Location of $y_{ijk.}$

$$= (i - 1)JK + (j - 1)K + (k - 1)1 + 1$$

If $y_{ijk.}$ is taken as the primary array, a secondary mapping in general with respect to this array is of the form

$$(i - 1)C_i + (j - 1)C_j + (k - 1)C_k + 1$$

where

$$C_\alpha = \begin{cases} 0 \text{ if the } \alpha\text{th subscript of the primary array does not occur in the secondary array,} \\ \text{The product of the limits of the subscripts appearing to the right of the } \alpha\text{th subscript in the secondary array or 1 if } \alpha \text{ is the last subscript in the secondary array otherwise.} \end{cases}$$

As an example of this secondary mapping, consider $y_{ij..}$, a secondary array with respect to $y_{ijk.}$. This mapping can be represented as

$$(i - 1)J + (j - 1)1 + (k - 1)0 + 1.$$

Utilization of the mappings presented provides the framework of combining secondary arrays with a given primary array.

The sign, S, which is employed to combine a secondary array with a primary one is given by

$$S = (-1)^{N_1+N_2}$$

where N_1 = the number of subscripts in the primary array,
N_2 = the number of subscripts in the secondary array.

5. Sums of Squares and Degrees of Freedom

Sums of squares and degrees of freedom for the analysis of variance model are computed in the following manner:

(a) The total sum of squares for the variable being processed is calculated as it is defined. For the example, this is

$$\sum_i \sum_j \sum_k \sum_l (y_{ijkl} - y....)^2.$$

(b) The model terms are operated upon term by term. For each term the residuals and degrees of freedom are formed. The residuals are squared, summed over all subscripts present in the term, and multiplied by the product of the limits of the subscripts not present to form the sum of squares. For the model term $TA(IJK)$ in the example, the sum of squares is computed as

$$L\sum_i \sum_j \sum_k (y_{ijk.} - y_{ij..} - y_{i.k.} + y_{i...})^2.$$

(c) The error sum of squares is obtained by subtracting the cumulative sums of squares associated with the model terms from the total sum of squares.

(d) The degrees of freedom are calculated as illustrated in Table I from the array of limits of subscripts.

Notice that all sums of squares other than the error sum of squares are formed from residuals. One of the principal reasons for taking this approach is the well-known fact that more accurate results are produced using this method as compared with other possible methods of computation.

Variance Components and F Values

In models containing random factors the estimation of the components of variation and testing hypotheses concerning their magnitude are primary objectives. In a model in which all factors are fixed, the calculation of F values presents no particular problem. However the analysis of models containing random factors requires the derivation of the expected mean square (EMS) of each source of variation. Rules for forming EMS's for analysis of variance models are presented in [3].

The composition of the EMS's of the model terms tells one how to perform the F test of the hypothesis corresponding to each line of the analysis of variance table. The numerator mean square used to test a certain hypothesis is the one corresponding to that line, while the denominator mean square employed is the one which has the same expected value as the numerator under the hypothesis. If for the denominator no such line exists, a linear combination of the estimates of the variance components is employed whose expectation equals that of the numerator mean square under the null hypothesis. If no line in the analysis of variance table is equal to the EMS of the numerator under the hypothesis, the appropriate F test, as described in [2], is only approximate. In this case an approximation of the degrees of freedom corresponding to the denominator must also be made. Tests in which the calculated mean square corresponding to one of the model terms can appropriately serve as the denominator for calculating the F value are termed exact tests (under the normality assumption). In exact tests the degrees of

TABLE III. EXPECTED MEAN SQUARES OF A MIXED MODEL

Source of variation	*Degrees of freedom*	*Expected mean square*
$P(I)$	$I - 1$	$\sigma^2 + L\sigma^2_{TA} + JL\sigma^2_{PA} + KL\sigma^2_{T} + JKL\sigma^2_{P}$
$T(IJ)$	$I(J-1)$	$\sigma^2 + L\sigma^2_{TA} + KL\sigma^2_{T}$
$A(K)$	$K - 1$	$\sigma^2 + L\sigma^2_{TA} + IJL\sigma^2_{A}$
$PA(IK)$	$(I-1)(K-1)$	$\sigma^2 + L\sigma^2_{A} + JL\sigma^2_{PA}$
$TA(IJK)$	$I(J-1)(K-1)$	$\sigma^2 + L\sigma_{TA}$
$E(IJKL)$	$IJK(L-1)$	σ^2
Total	$IJKL - 1$	

freedom of the denominator are the degrees of freedom of the model term whose mean square is being used as the denominator. In both exact and approximate F tests, the degrees of freedom of the numerator are the degrees of freedom corresponding to the source of variation being tested.

An algorithm for computing the variance component and F value corresponding to each source of variation is now discussed. This algorithm is a logical extension of what has been described up to this point and many of the arrays previously built are applicable. Table III gives the EMS's of the model presented earlier. A parameter (call it IFOR) conveniently indicates the random nature of these factors by a "1" in the appropriate positions of the word. For this model IFOR = 0110000000.

An important operation upon which the algorithm depends is the ordering of the model terms such that to obtain the estimate of the variance component and the denominator of the F value corresponding to a particular model term, one need only look at the model terms below the one being operated upon. If powers of two are assigned to the subscripts in the order in which they occur in the model, the desired rearrangement is accomplished if the numerical values of the subscripts of each of the model terms are summed, and the model terms rearranged such that a model term whose subscripts sum to less than the subscripts of another model term precedes it in the rearrangement.

Basically the procedure involves the determination of EMS's given the structure of the model and the random factors. Estimates of the variance components and the denominators necessary to compute F values for the model terms are computed term by term beginning with the term which contains the largest subscript sum. Estimates of variance components already computed at a given period in time are used to obtain values yet to be computed.

The steps performed to obtain the variance component and denominator for a model term in general (the Ith term) follow. The objective is to determine the composition of the EMS of the Ith term. Consider the Jth term as being one of the terms below the Ith after the rearrangement process.

(1) A "control word" is constructed for the Ith model term. It consists of a LOGICAL OR of the parameter $IFOR$ and the location of the $LMEFT$ array corresponding to the Ith model term.

(2) An inspection process is performed to determine if all of the subscripts which occur in the Ith term occur in the Jth term utilizing the pertinent locations of the $LMSUB$ array. If all of the subscripts do occur in the Jth term, the variance component of the Jth term is eligible for inclusion.

(3) (Assume the Jth term is eligible.) A comparison is made between $LMEFT(J)$ and the control word. If for each 1 which occurs in $LMEFT(J)$, a 1 also occurs in the corresponding position of the control word, then the variance component of the Jth term and its coefficient occur in the expected mean square of the Ith term.

Steps (2) and (3) are performed for each model term below the Ith after the rearrangement.

Consider the computations performed for the model term $T(IJ)$. At this stage of the algorithm the values $\hat{\sigma}^2$, $\hat{\sigma}^2_{TA}$, $\hat{\sigma}^2_{PA}$, and $\hat{\sigma}_A{}^2$ have been obtained.

(1) The control word is formed for $T(IJ)$:

$$IFOR = 0110000000$$

The effect symbols for $T(IJ)$ are:

0100000000

Thus the "control word" is:

0110000000

(2) $\hat{\sigma}^2$ appears in the denominator of the linear combination used for testing $\hat{\sigma}_T{}^2 = 0$ and in solving for its estimate.

(3) Next one determines if σ^2_{TA} appears in the EMS of $T(IJ)$. All of the subscripts contained in $T(IJ)$ are present in $TA(IJK)$. Also for each 1 in the location of $LMEFT$ corresponding to $TA(IJK)$ a 1 also appears in the control word.

control word.....0110000000
effect symbols of $TA(IJK)$.....0110000000

Thus $\hat{\sigma}^2_{TA}$ and its coefficient L appear in the EMS of $T(IJ)$.

(4) The model term $PA(IK)$ is now inspected. It is not included in the computations since all the subscripts in $T(IJ)$ are not in $PA(IK)$. (Also the factor P is fixed.)

(5) $\sigma_A{}^2$ is also not involved in the calculations for $T(IJ)$ since the subscript K does not occur in the model term $T(IJ)$.

The denominator for testing the hypothesis $\sigma_T{}^2 = 0$ has now been determined to be MS_{TA}, the mean square with expectation $\sigma^2 + L\sigma^2_{TA}$. The estimate of $\sigma_T{}^2$ is calculable from the equation

$$MS_T = \hat{\sigma}^2 + L\hat{\sigma}^2_{TA} + KL\hat{\sigma}_T{}^2.$$

By an inspection of Table III, clearly there exists no exact test for Ho: $\sigma_P{}^2 = 0$. In performing the computations for $P(I)$ the algorithm proceeds in its general manner

as before and computes

$$\hat{\sigma}^2 + L\hat{\sigma}^2_{TA} + JL\hat{\sigma}^2_{PA} + KL\hat{\sigma}_T{}^2$$

for testing this source of variation. Thus instances in which exact tests exist for certain model terms can be considered a special case of the algorithm.

If no exact F test exists for a certain model term, the degrees of freedom corresponding to the denominator used in calculating the F statistic for this term can be obtained as

$$\hat{\nu} = \frac{\hat{\gamma}^2}{\sum(\hat{\gamma}_i{}^2/\nu_i)}$$

$\hat{\gamma}$ is the value used as the denominator in calculating the F value, while the $\hat{\gamma}_i$ and ν_i are the calculated mean squares and degrees of freedom, respectively of the model terms whose EMS's when combined equal the expectation of γ. See [3] for a derivation of this formula. All values for calculating $\hat{\nu}$ are known besides the $\hat{\gamma}_i$ and these are obtained by constructing from the algorithm a table denoting the structure of the EMS's of the model terms. From the table a triangular set of equations solvable by a backward solution is easily obtained. The solution vector gives the calculated mean squares which are to be used in computing $\hat{\nu}$.

All concepts and methodology presented in this paper have been fully implemented on the IBM 7074 while the authors were members of the Statistical Laboratory at Iowa State University. An attempt was made to avoid machine dependencies. As a consequence, the program has been readily converted to the IBM 360, Model 50 now in use at Iowa State University. It is intended that the algorithms developed will form part of a more extensive statistical computing system oriented toward algebraic problem specification.

Acknowledgment. The authors wish to thank E. J. Carney, Assistant Professor of Statistics, Iowa State University, for constructing a preliminary version of the variance component algorithm to handle factorial models.

RECEIVED DECEMBER 1965; REVISED JULY 1966

REFERENCES

1. HEMMERLE, W. J. Algebraic specifications of statistical models for analysis of variance computations. *J. ACM 11* (1964), 234–239.
2. SATTERTHWAITE, F. E. An approximate distribution of estimates of variance components. *Biometrics 2* (1946), 110–114.
3. SCHEFFÉ, HENRY. *The Analysis of Variance*. John Wiley, New York, 1959.
4. SCHLATER, J. E. Analysis of variance and covariance computations on a digital computer for balanced complete structures based on algebraic model specifications. M.S. Thesis, Iowa State U. Library, 1965.

Algorithms

J. G. HERRIOT, Editor

ALGORITHM 293
TRANSPORTATION PROBLEM [H]

G. BAYER (Recd. 9 July 1965 and 22 Aug. 1966)
Technische Hochschule, Braunschweig, Germany

```
procedure transp1 (m, n, inf, c, a, b, x, kw); value m, n, inf;
  integer m, n, inf, kw; integer array c, a, b, x;
comment transp1 is derived from Algorithm 258, transport,
  [Comm. ACM 8 (June 1965), 381] in order to reduce running time
  by about 50 percent. The following notation is used.
  c   m, n-matrix of unit costs,
  a   array of quantities available,
  b   array of quantities required, following the usual description of the transportation problem,
  inf greatest positive integer within machine capacity,
  x   m, n-matrix of flows,
  kw  optimal total costs (computed by procedure).
  c, a, b are disturbed by the procedure. Sum of a[i] = sum of b[i].
  Multiple solutions are left out of account. [Ref.: G. Hadley,
  Linear Programming, Reading, London, 1962, p. 351];
begin integer i, j, u, v, k, l, s, t, gd, h, p, cij, xij, ai, bj, lsvj, nlvi;
  Boolean zg;
  integer array g, listu, nlv[1:m], r, listv[1:n], ls[0:m+n-1],
    nl[1:m×n], lsv[0:n];
  comment in the for-statement u := ··· after s33, operate on
    all pairs i, j with c[i,j] = 0. To win time the array nl supervises
    those zeros; the j-indices of zeros in row i are kept in
    nl[(i-1)×n+1] ··· nl[nlv[i]]. In the for-statement v := ···
    after s33, operate on all pairs i, j with x[i,j] ≠ 0 (and c[i,j]=0).
    ls supervises those essential zeros, the i-indices of essential
    zeros in column j are kept in ls[lsv[j-1]+1] ··· ls[lsv[j]
    Procedure in adds to list ls, procedure out takes out from list
    ls an essential zero in position i, j;
  procedure in;
  begin
    lsvj := lsv[j];
    for t := lsv[n] step -1 until lsvj do ls[t+1] := ls[t];
    for t := j step 1 until n do lsv[t] := lsv[t] + 1;
    ls[lsvj+1] := i
  end ;
  procedure out ;
  begin
    lsvj := lsv[j];
    for t := lsv[j-1]+1 step 1 until lsvj do
    begin
      if ls[t] ≠ i then go to next;
      s := t; go to ex;
  next:
    end ;
  ex:
    for t := j step 1 until n do lsv[t] := lsv[t]-1;
    lsvj := lsv[n];
    for t := s step 1 until lsvj do ls[t] := ls[t+1]
  end ;
  for i := 1 step 1 until m do
    for j := 1 step 1 until n do x[i,j] := 0;
  for i := 1 step 1 until m do nlv[i] := (i-1)×n;
```