

A Generalization of the Stuart-Maxwell Test

Stephen Black and Mithat Gönen
Applied Logic Associates, Inc.

1. Introduction

A variable which can take only a small number of different values (categories), is called a categorical variable. Some variables are categorical by their nature, such as gender, presence or absence of a medical condition, answers to multiple choice questions in a survey, etc. On the other hand, it may be convenient to analyze a non-categorical variable (such as age) using a few categories (less than 25, 25-45, more than 45) in some cases. In either situation, there are a variety of methods available at our disposal to analyze categorical data. Texts by Fleiss (1981) and Agresti (1990) provide a good overview of such methods.

When two categorical variables are measured on the same subject, it is useful to visualize their relationship using a contingency table. As an example, consider the following hypothetical data, where a particular clinical measurement was taken twice on each of 13 patients, once before each patient received treatment and again after treatment had been administered. The measurements are categorized into Low, Normal and High values, based on standard ranges of values for this particular measurement. The data is summarized in the contingency table below:

Table 1

Before Treatment	After Treatment			Total
	Low	Normal	High	
Low	4	0	0	4
Normal	0	36	0	36
High	0	8	4	12
Total	4	44	4	52

The numbers in the table are interpreted as follows: 4 of the patients had a measurement considered Low before treatment was administered and another considered Low after treatment had been received, whereas 8 patients had values considered High before treatment was received and measurements considered to be Normal after treatment administration. Similar interpretations hold for the other numbers in the table.

One question of interest in such a situation is whether the probability distribution for the classification of the patients' measurements is the same before and after treatment administration. In other words, "What is the probability that the pre-treatment measurement and the post-treatment measurement fall into the same category?" If that is the case, then the contingency table is said to display marginal homogeneity. Several real-life examples where marginal homogeneity is studied are given by Krauth (1986).

A test for marginal homogeneity is proposed by Stuart (1955) and Maxwell (1970). The way the test procedure is developed in these two articles requires a certain matrix related to the contingency table to be invertible. We have found in our experience that this is not always the case in practice. To solve the problem we develop a generalization of the Stuart-Maxwell test, which will work even when the mentioned matrix is not invertible. This paper is an attempt to explain our methodology and SAS® code using PROC IML that carries out the generalization of the Stuart-Maxwell test.

In Section 2, we will give a brief account of the Stuart-Maxwell test and describe the matrix that needs to be inverted. Section 3 will explain our methodology to avoid the problems of the current approach and demonstrates the difference between the Stuart-Maxwell test and its generalization using an example. Section 4 presents a discussion and concluding remarks.

2. Stuart-Maxwell Test

Consider Table 1 that we used to display the data from the two measurements for each patient in our hypothetical clinical trial. Let p_{ij} denote the probability of a patient's first measurement being classified into the i^{th} category and the second measurement into the j^{th} category. For example, p_{23} will denote the probability of the pre-treatment measurement being classified as Normal and the post-treatment one as High. Let us define $p_{i.} = \sum_j p_{ij}$ and $p_{.j} = \sum_i p_{ij}$. These quantities are called marginal probabilities. For example $p_{2.}$ is the probability that the pre-treatment measurement will be classified as Normal and $p_{.1}$ is the probability that the post-treatment value will be classified as Low. To see if the two assessment times have the same probability distribution of classification, we can formulate the following hypothesis: $H_0: p_{i.} = p_{.i}$ for $i = 1, 2, 3$. If we fail to reject the null hypothesis, we will say that the clinical assessments do not differ significantly in their classifications from pre-treatment to post-treatment.

Let n_{ij} denote the number of patients with pre-treatment measurements classified to the i^{th} category and post-treatment values to the j^{th} category. They are simply the entries in our contingency table. Similar to our notation in probabilities, let $n_{i.}$ and $n_{.j}$ denote the row and column totals in the table. Finally, let n be the sample size, which is the total of the row totals (or equivalently column totals) in the table.

A starting point for the analysis is to examine how the two distributions differ between the assessment times. This can be achieved by looking at the respective differences in their classifications. Specifically, if we define $d_i = n_{i.} - n_{.i}$, we can gain some insight by investigating the d_i 's. It is important to note that, by definition, $\sum_i d_i = 0$. In statistical terminology, we say that d_i 's are linearly dependent. Stuart (1955) showed that

$$\begin{aligned}\hat{V}_{ij} &= n_{i.} + n_{.i} - 2n_{ii} \\ \hat{V}_{ij} &= -(n_{ij} + n_{ji})\end{aligned}\tag{1}$$

where V_{ii} is the variance of d_i and V_{ij} is the covariance between d_i and d_j . The $\hat{}$ signs above the V 's indicate that these formulas give us an estimate of the variances and covariances. Note that it is natural to present the V_{ij} 's in a matrix which is simply called the covariance matrix and we will denote it with V . In linear algebra, if a matrix V^{-1} exists such that $V V^{-1} = I$, where I is the identity matrix, then we say V is invertible and V^{-1} is said to be the inverse of V . It turns out that, since the d_i 's are linearly dependent, the matrix V is singular (not invertible).

Current methodology proposes excluding some elements of d_i until it becomes linearly independent and then considering the corresponding covariance matrix. Let us assume that we were able to reduce the matrix V to a dimension where it is invertible and call its inverse W . Then, Stuart (1955) showed that

$$Q = \sum_{i,j=1}^3 W_{ij} d_i d_j\tag{2}$$

is an appropriate test statistic to use for marginal homogeneity. In particular he showed that, if n is large enough, we can approximate Q using a χ^2 distribution.

Usually, it will be enough to eliminate one of the d 's; however, in some cases where the table is sparse, most of the off-diagonal elements of V will be 0 (see the formula for V_{ij}), and we will need further elimination. Such elimination does not lend itself to an easy algorithmic presentation and hence is extremely complicated to program. We present an alternative approach to this problem in the next section where we make use of a mathematical tool called a "generalized inverse" which enables us to avoid any kind of elimination and results in a simple and efficient algorithm.

3. Generalization of the Stuart-Maxwell Test

The main tool that we will use to generalize the Stuart-Maxwell test is the generalized inverse. As a review, the inverse of a square matrix V is any matrix V^{-1} satisfying the condition $VV^{-1} = V^{-1}V = I$. We have already pointed out that such a matrix does not always exist; however, there always exists a matrix V^- called "generalized inverse of V " which satisfies $VV^-V = V$ and $V^-VV^- = V^-$. These generalized inverses possess some of the useful properties of the regular inverses. It is also reassuring to know that if a regular inverse exists it is the same as the generalized inverse. In other words, for any matrix V , we can find a generalized inverse V^- , but if it also happens that a regular inverse V^{-1} exists, then $V^{-1} = V^-$.

Now consider Q as given in (2). A compact representation of Q can be given in terms of vectors and matrices. If we represent d_i with a column vector d , then Q can also be written as $d'Wd$, where W is the inverse of V . Such representations are known as quadratic forms and are very common in statistics. For example, Stuart's result is a special case of Wald's class of test statistics (1943) which deal with the distribution of quadratic forms when the elements of d in a quadratic form are random variables. If W is an n -dimensional matrix whose inverse exists and the elements of d are normally distributed random variables with covariance matrix V , then Q is known to have a χ^2 distribution with n degrees of freedom.

An important concept about a matrix is its rank. The rank of a square matrix is the number of linearly independent columns (or, equivalently, rows) that it contains. We have already seen that rank must be equal to the dimension for a square matrix in order for an inverse to exist (sometimes such matrices are called full-rank). In light of the definition of rank and the discussion about the distribution of quadratic forms in the previous paragraph, the following fact which was established by Rao and Mitra (1971, p.173) should not come as a surprise: "Let d be a vector whose elements are normally distributed random variables with covariance matrix V , then the quantity $Q = d'V^-d$ has a χ^2 distribution with k degrees of freedom where k is the rank of V ." This is essentially a generalization of Wald's result that was discussed in the previous paragraph. When V is full-rank, the result of Rao and Mitra (1971) reduces to that of Wald (1943).

We can easily use this result to generalize the Stuart-Maxwell test. There is no need to eliminate the rows and columns in search of an invertible submatrix of V . We already know that $d'V^-d$ has a χ^2 distribution with k degrees of freedom where k is the rank of V . All we need to do is calculate the value of $d'V^-d$ and find the rank k of V . Then we can compare this with the quantiles from a χ^2 distribution with k degrees of freedom and obtain a p -value for the statistics. If V happened to be a matrix which adhered to Stuart's suggestion of elimination of one row or column, our method gives us the same result automatically, since the result of Rao and Mitra reduces to that of Wald for full-rank matrices. When we need more than one elimination, we do not need to resort to complicated algorithms, because the features of the generalized inverse take care of this for us.

Generalized inverses are not unique. That is, for a given matrix there are many matrices which satisfy the conditions for a generalized inverse. A way to make it unique is to impose further restrictions on such matrices so that one and only one can satisfy the specified condition. A popular set of such conditions is known as the Moore-Penrose conditions and the resulting generalized inverse is known as the Moore-Penrose inverse. PROC IML has a built-in function called GINV to calculate the Moore-Penrose inverse. Since, for our purposes, it does not matter which generalized inverse we use, we choose the Moore-Penrose inverse as it is readily available in SAS®. There is no built-in function in PROC IML to calculate the rank of a matrix, but it can easily be calculated using the following code in PROC IML:

```
MAT_RANK=NCOL(LOC(T(EIGVAL(MAT_NAME))))
```

where MAT_NAME is the name of the matrix that we are working with and MAT_RANK is its rank. This calculation uses the fact that the rank of a matrix is equal to the number of non-zero eigenvalues.

The full SAS® code that performs the generalized Stuart-Maxwell test can be found in the Appendix.

Now consider the example from Section 1. We can calculate that

$$\mathbf{d} = (0 \quad 8 \quad -8)$$

$$\mathbf{V} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 8 & -8 \\ 0 & -8 & 8 \end{bmatrix}$$

where it is easy to see that Stuart's recommendation of eliminating the last column and row does not work, since it would leave us with the following matrix:

$$\begin{bmatrix} 0 & 0 \\ 0 & 8 \end{bmatrix}$$

which is still not invertible. In fact, this covariance matrix is of rank 1 and we need further reduction. However using the generalized inverse leads to

$$Q = (0 \quad 8 \quad -8) \begin{bmatrix} 0 & 0 & 0 \\ 0 & 8 & -8 \\ 0 & -8 & 8 \end{bmatrix}^{-} \begin{pmatrix} 0 \\ 8 \\ -8 \end{pmatrix} = 8$$

which gives us a p -value of 0.0047 when compared with a χ^2 distribution with 1 degree of freedom. Thus we reject the null hypothesis of marginal homogeneity and conclude that the classifications of patients differ significantly between pre-treatment and post-treatment measurements.

4. Conclusions

Testing for marginal homogeneity in square contingency tables using the Stuart-Maxwell test is a common procedure in many areas of statistics. The methods that are commonly in use might fail in the presence of singular covariance matrices when the contingency table is sparse. In this article, we presented a method using the generalized inverses which avoids this problem. Our method has wide applicability and is easy to program as demonstrated by the attached SAS[®] code.

Note: All data is fabricated and are for demonstration purposes only.

SAS is a registered trademark of SAS Institute, Inc. in the USA and other countries.

[®] indicates USA registration.

Please address all the correspondence to:

Applied Logic Associates
Stephen Black
Manager, Biostatistical Programming
5615 Kirby Drive, Suite 710
Houston, TX 77005-2495
E-MAIL: sblack@alogic.com

References

- Agresti, A. (1990). *Categorical Data Analysis*, John Wiley and Sons, New York.
- Fleiss, J. L. (1981). *Statistical Methods for Rates and Proportions*, 2nd edn., John Wiley and Sons, New York.

- Krauth, J. (1985). A Comparison of Tests for Marginal Homogeneity in Square Contingency Tables. *Biometrical Journal*, 1:3-15.
- Maxwell, A. E. (1970). Comparing the Classification of Subjects by Two Independent Judges. *British Journal of Psychiatry*, 116:651-655.
- Rao, C. R. and Mitra, S. K. (1971). *Generalized Inverse of a Matrix and Its Applications*. John Wiley and Sons, New York.
- Stuart, A. (1955). A Test for Homogeneity of the Marginal Distributions in a two-way Classification. *Biometrika*, 42:412-416.
- Wald, A. (1943). Tests of Statistical Hypotheses Concerning Several Parameters When the Number of Observations is Large. *Transactions of the American Mathematical Society*, 54: 426-482.

Appendix

```

*****;
* Client: All Clients of Applied Logic Associates, Inc. *;
* *;
* Protocol: All Protocols *;
* *;
* Program: STUDAT01.SAS *;
* *;
* Input Files: User-specified Dataset *;
* *;
* External Macros: %SORT *;
* *;
* Output Files: User-Specified Dataset that Contains Values of all the Test *;
* Statistics which are Calculated in the Program *;
* *;
* Tables: None *;
* *;
* Programmer: MBELL *;
* *;
* Date Completed: 11NOV93 *;
* *;
* Description: Program calculates the Stuart-Maxwell or McNemar test *;
* statistic (which ever is appropriate) for use with *;
* transition tables for categorical variables. *;
* *;
* Revisions: 27JUL95 JFULTON and SBLACK generalized macro so it can run *;
* without protocol-specific adjustments *;
* 02AUG95 SBLACK updated program to say NO TEST/NO DIFFERENCES* *;
* whenever the Statistic Could not be calculated due *;
* to no change from BL instead of just displaying a *;
* missing value for the statistic *;
* 08AUG95 SBLACK added Validation Documentation Line *;
* 10AUG95 SBLACK changed the phrase NO TEST/SPARSE DATA to the* *;
* phrase NO TEST POSSIBLE for those instances where *;
* the matrix reduced to smaller than a 2 X 2 *;
* 01SEP95 SBLACK updated program to make it consistent with *;
* the other standard macros (no code changes) *;
* 28SEP95 SBLACK updated program to use the standard macro *;
* header specified in SAS_HEAD.SAS (follows the header* *;
* specified in the Biostatistics SOPs) *;
* 04OCT95 SBLACK updated program to have USERIDs (not names) *;
* 14APR96 ARODGERS made macro variable names consistent with *;
* the Standard Macro Variables and changed name of *;
* macro to studat01 to be the same as the name of the *;
* program *;
* 23MAY96 JFULTON generalized further by updating header, *;
* replacing BY &BYVAR with just &BYVAR, creating *;
* &BYVAR2 to be used in KEEP statements, adding _ to *;
* beginning of data set or var names created in macro,* *;
* 23APR97 SBLACK updated program to calculate the statistic *;
* based on the generalized inverse of the covariance *;
* matrix instead of the true inverse in cases where *;

```

```

*               the reduced dimension covariance matrix is not of *;
*               full rank and the true inverse cannot be calculated.*;
*
* Validation:      User_ID      Date      Validation
*                 ARODGERS      21MAY96   Generalization
*                 ARODGERS      21MAY96   Approval
*                 JFULTON       28MAY96   Macro Change
*                 SBLACK        23APR97
*
*-----*
* DESCRIPTION OF MACRO PARAMETERS SENT FROM YOUR PROGRAM:
*
*   _BASE:  BASE is the NAME of the variable used in your program to indicate
*           the values of the parameter at baseline.
*
*   _BYVAR: _BYVAR is the NAME of the variable used to indicate BY variables
*           for both sorting and setting datasets
*
*   _DATDEL: Specify YES/NO (Default=YES) to indicate whether all work datasets
*            created by the macro should be deleted at the end of the macro
*
*   _DIM:   DIM is the dimension of the transition table, i.e.-the total
*           number of possible values of the variables represented by BASE and
*           PARAM
*
*   INDSN:  Initial data set to be sent to the macro.
*
*   OTDSN:  This is the data set which holds the values of the byvariables
*           and the test stat, p-value, etc. which you will want to print out
*
*   _PARAM: PARAM is the NAME of the variable used in your program to
*           indicate the values of the parameter at follow-up.
*
*   _TREAT: TREAT is the NAME of the variable which the data set you send to
*           the macro uses to specify treatment values.
*
*-----*
* RESTRICTIONS:
*
*   1) BE SURE YOU HAVE ACCURATELY COMPLETED THE LIBNAMES.SAS MACRO BEFORE
*      ATTEMPTING TO USE THIS ONE. AND %INCLUDE LIBNAMES AT THE TOP OF YOUR
*      PROGRAM.
*
*   2) THE VALUES OF THE PARAMETER BEING ANALYZED MUST RANGE FROM 1 TO THE
*      MAX POSSIBLE VALUE. EX: IF A LAB PARAMETER CAN BE 0,1,OR 2 FOR LOW,
*      NORMAL, HIGH, RE-FORMAT TO 1,2,OR 3 FOR LOW, NORMAL, HIGH,
*      RESPECTIVELY.
*
*   3) DO NOT INCLUDE ANY DUMMY VARIABLES IN THE DATA SET SENT TO THIS MACRO.
*
*   4) DO NOT LET OTDSN = ANY OF THE FOLLOWING: _STUMAX, _STUFREQ, _STUINV,
*      _STUINV1, _STUINV2, _STUMAX1, _STUNV2A, _STUNV2B, _STUNV2C, _MATRX,
*      _FINMAT, _FINDET, _FINRANK, _GENINV, _DET, _RANK, _EXTRA. AS THESE
*      DATASETS ARE DELETED AFTER EXECUTION OF THIS MACRO IF _DATDEL = YES.
*
*   5) VALUES OF _VALUE VARIABLE SHOULD BE FORMATTED WITH A FORMAT CONTAINING
*      -1 = 'EXACT TEST' BEFORE PRINTING OUT THE RESULTING OUTPUT DATASET
*
*-----*
%MACRO STUDAT01 (INDSN=,
                _BASE=,
                _PARAM=,
                _DIM=,
                _TREAT=TREAT,
                _BYVAR=%STR(BY ),
                OTDSN=,
                _DATDEL=YES);
*-----*

%LET _BYVAR2 = %SUBSTR(&_BYVAR,4);

%SORT(INDSN=&INDSN, OTDSN=_STUINIT, _BYVAR=&_BYVAR &_TREAT) ;

/**** Initialize Output STU_MAX Dataset ****/

DATA _STUMAX;
  SET _NULL_;
  RUN;

```

```

/*****
** BEGIN CALCULATION OF TEST STATISTICS
**
*****/
%LET DIMSQ = %EVAL(&_DIM**2) ;
%LET DIMPL = %EVAL(&_DIM*2) ;

PROC FREQ DATA = _STUINIT;
  &_BYVAR &_TREAT;
  TABLES &_BASE * &_PARAM/OUT = _STUFREQ (DROP = PERCENT) NOPRINT;
RUN;

%SORT(INDSN=_STUFREQ, _BYVAR=&_BYVAR &_TREAT &_BASE &_PARAM);

DATA _STUINV ;
  SET _STUFREQ ;
  &_BYVAR &_TREAT &_BASE &_PARAM ;

  ARRAY A(&_DIM,&_DIM) A1-A&DIMSQ ;

  /*** INITIALIZE ARRAY TO MISSING */

  IF FIRST.&_TREAT THEN DO ;
    DO _I = 1 TO &_DIM ;
      DO _J = 1 TO &_DIM ;
        A(_I,_J) = . ;
      END ;
    END ;
  END ;

  /*** FILL IN ARRAY WITH COUNTS FROM PROC FREQ, ZERO OUT MISSING VALUES */

  DO _I = 1 TO &_DIM ;
    DO _J = 1 TO &_DIM ;
      IF &_BASE = _I AND &_PARAM = _J THEN A(_I,_J) = COUNT ;
      IF A(_I,_J) = . THEN A(_I,_J) = 0 ;
    END ;
  END ;

  RETAIN A1-A&DIMSQ ;

  IF LAST.&_TREAT ;

  KEEP &_BYVAR2 &_TREAT A1-A&DIMSQ;
RUN ;

DATA _STUINV1 ;
  SET _STUINV ;
  &_BYVAR &_TREAT;

  ARRAY A(&_DIM,&_DIM) A1-A&DIMSQ ;
  ARRAY R(&_DIM) R1-R&_DIM ;
  ARRAY C(&_DIM) C1-C&_DIM ;

  IF FIRST.&_TREAT THEN _CNT = 0 ;

  /*** INITIALIZE TOTAL VARIABLES FOR ROWS AND COLUMNS */

  DO _I = 1 TO &_DIM ;
    R(_I) = 0 ;
    C(_I) = 0 ;
  END ;

  /*** CALCULATE ROW AND COLUMN TOTALS */

  DO _I = 1 TO &_DIM ;
    DO _J = 1 TO &_DIM ;
      R(_I) = R(_I) + A(_I,_J) ;
      C(_I) = C(_I) + A(_J,_I) ;
    END ;
  END ;

```

```

    /*** MATRIX REDUCTION SECTION TO PREVENT SINGULARITY ***/
    /*** IF CORRESPONDING ROWS AND COLUMNS ARE ZERO, *****/
    /*** SET THEIR ELEMENTS TO MISSING. *****/

    DO _I = 1 TO &_DIM ;
        IF C(_I) = 0 AND R(_I) = 0 THEN DO ;
            DO _J = 1 TO &_DIM ;
                A(_I,_J) = . ;
                A(_J,_I) = . ;
            END ;
            _CNT+1 ;
        END ;
    END ;

    /*** RESTRUCTURE THE ARRAY ELEMENT LIST SO THAT THE ELEMENTS OF THE ;
    *** ZERO ROWS & COLUMNS ARE REMOVED FROM THEIR PRESENT POSITION AND ;
    *** THE REMAINING ELEMENTS SLIDE DOWN TO TAKE THEIR PLACE. THIS ;
    *** ALLOWS ONE TO ADJUST ARRAY DIMENSION SIZE AND PICK THE CORRECT ;
    *** ELEMENTS. *****/

    IF _CNT NE 0 THEN DO ;
        _I2 = 1 ;
        _J2 = 1 ;

        DO _I = 1 TO &_DIM ;
            DO _J = 1 TO &_DIM ;
                IF A(_I,_J) NE . THEN DO ;
                    A(_I2,_J2) = A(_I,_J) ;
                    _J2+1 ;
                    IF MOD(_J2,&_DIM) = 1 THEN DO ;
                        _J2 = 1 ;
                        _I2+1 ;
                    END ;
                END ;
            END ;
        END ;

        /*** CREATE VARIABLE WHICH HOLDS ADJUSTED ARRAY DIMENSION **/
        NEW = &_DIM - _CNT ;

    RUN;

    /*** Find Maximal Value of NEW to Control Loop ***
    *** and Make it a Macro Variable *****/
    PROC MEANS DATA=_STUINV1 MAX NOPRINT;
        VAR NEW;
        OUTPUT OUT=_STUMAXI MAX=MAX;
    RUN;

    DATA _NULL_;
        SET _STUMAXI;
        CALL SYMPUT('NEWMAX', MAX);
    RUN;

    %SORT(INDSN=_STUINV1, OTDSN=_STUINV, _BYVAR=BY NEW) ;

    /*** Loop Over All Possible Values of NEW to Calculate the Test Statistics ***/

    %DO NEW = 0 %TO &NEWMAX; /** Master Loop for NEW Values **/

    /*** Only Calculate Statistic for Values of NEW That Have Data ***/
    %LET _FLAG = 0;

    DATA _STUINV1;
        SET _STUINV;
        BY NEW;
        WHERE NEW = &NEW;
        IF FIRST.NEW AND NEW ^= . THEN CALL SYMPUT('_FLAG',1);
    RUN;

    %IF &_FLAG = 1 %THEN %DO; /** Inner Loop for NEW Values with Data **/

    /*** CREATE MACRO VARIABLES NEEDED FOR MATRIX OPERATIONS **

```

```

*** TO FOLLOW ACCORDING TO NEW(REDUCED) ARRAY DIMENSION **/

%LET DIM2 = %EVAL(&NEW) ;
%LET DIMSQ2 = %EVAL(&NEW**2) ;
%LET DIMPL2 = %EVAL(&NEW*2) ;

/**** MCNEMAR STATISITIC CALCULATIONS FOR 2x2 TABLES,
*** STUART-MAXWELL CALCULATIONS FOR KxK TABLES (K>2) */

%IF &NEW = 2 %THEN %DO ; /** IF NEW = 2 **/

DATA _STUINV2 ;
LENGTH STATTYPE $25;
SET _STUINV1 ;

_DENOM = A2 + A3 ;
IF _DENOM=0 THEN DO;
STATTYPE = 'NO TEST/NO DIFFERENCES';
GENFLAG = .;
_VALUE = . ;
_PVALUE = .;
END;

/**** USE NORMAL APPROXIMATION CALCULATION *** ;
*** IF # OF DISCORDANT PAIRS (_DENOM) >= 20 ***/

ELSE IF _DENOM GE 20 THEN DO ;
STATTYPE = 'MCNEMAR';
GENFLAG = .;
_VALUE = (ABS(A2 - A3) - 1)**2 / _DENOM ;
_PVALUE = 1 - PROBCHI(_VALUE, 1) ;
END ;

/**** USE EXACT BINOMIAL CALCULATION IF *** ;
*** # OF DISCORDANT PAIRS (DENOM) < 20 ***/

ELSE IF A2 LT (_DENOM/2) THEN DO ;
STATTYPE = 'MCNEMAR';
GENFLAG = .;
_VALUE = -1 ;
_PVALUE = 2 * PROBBNML(0.5, _DENOM, A2) ;
END ;

ELSE DO ;
STATTYPE = 'MCNEMAR';
GENFLAG = .;
_VALUE = -1 ;
_PVALUE = 2 * (1 - PROBBNML(0.5, _DENOM, (A2-1))) ;
IF _PVALUE > 0.9999 THEN _PVALUE = 0.9999 ;
END ;

IF _PVALUE LE 0.05 AND _PVALUE NE . THEN _SIGP='*' ;

RUN ;

%END ; /** IF NEW = 2 **/

%ELSE %IF &NEW > 2 %THEN %DO ; /** IF NEW > 2 **/

DATA _STUNV2A(DROP=_I _II _J _X _Y);
SET _STUINV1 ;

ARRAY NN(&DIMPL2) NN1-NN&DIMPL2 ;
ARRAY D(&DIM2) D1-D&DIM2 ;
ARRAY V(&DIM2,&DIM2) V1-V&DIMSQ2 ;
ARRAY A(&DIM2,&DIM2) A1-A&DIMSQ2 ;

NCHNG = 0;
DO _X = 1 TO &DIM2;
DO _Y = 1 TO &DIM2;
IF _X ^= _Y THEN NCHNG = NCHNG + A(_X,_Y);
END;
END;

```

```

DO _I = 1 TO &DIM2 ;
  _II = &DIM2 + _I ;
  NN(_I) = 0 ;
  NN(_II) = 0 ;
  DO _J = 1 TO &DIM2 ;
    NN(_I) = NN(_I) + A(_I,_J) ;
    NN(_II) = NN(_II) + A(_J,_I) ;
  END ;
END ;

DO _I = 1 TO &DIM2 ;
  _II = &DIM2 + _I ;
  D(_I) = NN(_II) - NN(_I) ;
END ;

DO _I = 1 TO &DIM2 ;
  _II = &DIM2 + _I ;
  DO _J = 1 TO &DIM2 ;
    IF _I EQ _J THEN V(_I,_J) = NN(_I) + NN(_II) - 2*A(_I,_J) ;
    IF _I NE _J THEN V(_I,_J) = -( A(_I,_J) + A(_J,_I) ) ;
  END ;
END ;

/**** CREATE FLAG VARIABLE TO DETERMINE IF THERE ARE CHANGES BETWEEN THE TWO ****/
/**** DISTRIBUTIONS (DISTRIBUTION OF ACROSS VARIABLE AND DISTRIBUTION OF DOWN **/
/**** VARIABLE) - CHECK FOR OFF-DIAGONAL VALUES. IF THERE ARE NO CHANGES (ALL**/
/**** OFF-DIAGONAL VALUES ARE ZERO) THEN NO TEST CAN BE PERFORMED *****/
CALL SYMPUT("NCHGFLG", NCHNG);
RUN;

/**** IF NO CHANGES THEN STATE THAT INSTEAD OF CALCULATING A STATISTIC. OTHERWISE, *****/
/**** CALCULATE THE APPROPRIATE STATISTIC BASED ON THE SINGULARITY OF THE COVARIANCE MATRIX **/
%IF &NCHGFLG=0 %THEN %DO; /**** NO OFF-DIAGONAL VALUES ***/
  DATA _STUINV2;
    LENGTH STATTYPE $25;
    SET _STUINV1;
    STATTYPE = 'NO TEST/NO DIFFERENCES';
    GENFLAG = .;
    _VALUE = .;
    _PVALUE = .;
    _SIGP = ' ';
  %END; /**** NO OFF-DIAGONAL VALUES ***/
%ELSE %DO; /**** OFF-DIAGONAL VALUES PRESENT ***/

/***** 22APR1997 SBLACK ADDED CODE TO CALCULATE GENERALIZED VERSION OF THE STUART-MAXWELL
*****/
/***** STATISTIC FOR CASES WHEN THE REDUCED DIMENSION COVARIANCE MATRIX IS SINGULAR AND
*****/
/***** THE TRUE INVERSE CANNOT BE CALCULATED. IN THIS CASE WE USE THE VALUES FROM THE
*****/
/***** GENERALIZED INVERSE OF THE COVARIANCE MATRIX AS THE WEIGHTS TO CALCULATE THE STATISTIC
**/

/***** NEED TO DETERMINE THE NUMBER OF BY-GROUP*TREATMENT COMBINATIONS *****/
/***** SO WE WILL KNOW HOW MANY SUBMATRICES WE WILL HAVE TO CALCULATE *****/
PROC SORT DATA=_STUNV2A;
  &_BYVAR &_TREAT;
RUN;

DATA _NULL_;
  SET _STUNV2A END=LAST;
  &_BYVAR &_TREAT;
  IF FIRST.&_TREAT THEN _MATCNT+1;
  IF LAST THEN CALL SYMPUT("_MATCNT",_MATCNT);
RUN;

/***** TAKE CURRENT DATASET WITH ONE RECORD AND &DIMSQ2 OBSERVATIONS AND CONVERT *****/
/***** IT INTO A MATRX-TYPE DATASET WITH &DIM2 OBSERVATIONS AND &DIM2 RECORDS *****/

DATA _MATRX(DROP=X Y A1-A&DIMSQ2 V1-V&DIMSQ2);
  SET _STUNV2A;

  ARRAY V (&DIMSQ2) V1 - V&DIMSQ2;
  ARRAY U (&DIM2) U1 - U&DIM2;

  DO X=0 TO &DIM2-1;
    DO Y = 1 TO &DIM2;

```

```

        U(Y) = V(X*DIM2 + Y);
    END;
    OUTPUT;
END;
RUN;

/***** CREATE LIST OF VARIABLES (&_VARLIST) TO BE USED AS THE COLUMNS AND *****/
/***** A LIST OF ROWS TO BE INCLUDED IN EACH SUBMATRIX OF THE LARGER MATRIX *****/
/***** WHICH DEFINES THE COVARIANCE MATRIX OF EACH COMBINATION OF BY VARIABLES **/
/***** SO THAT THE GENERALIZED INVERSE CAN BE CALCULATED FOR EACH COMBINATION **/
/***** OF BY VARIABLES SPECIFIED BY &_BYVAR AND &_TREAT. WE ONLY NEED THE *****/
/***** &DIM2-1 ROWS AND COLUMNS IN ORDER TO CALCULATE THE WEIGHTS NEEDED TO ****/
/***** CALCULATE THE STUART-MAXWELL STATISTIC *****/
%LET DIM2R = %EVAL(&DIM2-1);
%LET DIMSQ2R = %EVAL(%EVAL(&DIM2-1)**2);
%LET _VARLIST=;
%DO I = 1 %TO &DIM2R;
    %LET _VARLIST=&_VARLIST U&I;
%END;

%LET _TCNT=1;
%DO I = 1 %TO &_MATCNT;
    %LET _ROWLST&I=;
    %DO J = 1 %TO &DIM2;
        %IF &J<=&DIM2R %THEN %LET _ROWLST&I=&&_ROWLST&I &_TCNT;
        %LET _TCNT = %EVAL(&_TCNT+1);
    %END;
%END;

/**** USE PROC IML TO CALCULATE THE GENERALIZED INVERSE OF EACH COVARIANCE MATRIX **/
/**** AS WELL AS TO CALCULATE THE RANK AND DETERMINANT OF EACH MATRIX *****/
PROC IML;

    /**** INITIALIZE MATRIX TO HOLD COMBINED GENERALIZED INVERSE MATRICES **/
    /**** AND MATRIX TO HOLD DETERMINANTS AND ANOTHER TO HOLD THE RANKS *****/
    ALLG=J(&_MATCNT*&DIM2, &DIM2R, .);          /** GENERALIZED INVERSE VALUES **/
    ALLD=J(&_MATCNT, 1, .);                        /** DETERMINANTS **/
    ALLR=J(&_MATCNT, 1, .);                        /** RANKS **/

    /**** READ IN VALUES OF COMBINED COVARIANCE MATRICES TO CALCULATE GENERALIZED INVERSES ****/
    EDIT _MATRX;
    READ ALL VAR (&_VARLIST) INTO U;

    /**** CREATE A SEPARATE SUBMATRIX FOR EACH COVARIANCE MATRIX *****/
    /**** AND FIND THE GENERALIZED INVERSE OF EACH AND THEN COMBINE ****/
    /**** GENERALIZED INVERSES TOGETHER INTO ONE MATRIX FOR OUTPUT *****/
    %DO I = 1 %TO &_MATCNT;
        SUBMAT&I=U[(&&_ROWLST&I),];
        G&I=GINV(SUBMAT&I);
        ALLG[(&&_ROWLST&I),]=G&I;
        ALLD[(&I),]=DET(SUBMAT&I);

        /**** THE RANK OF A MATRIX IS EQUAL TO THE NUMBER OF NON-ZERO EIGENVALUES ****/
        ALLR[(&I),]=NCOL(LOC(T(EIGVAL(SUBMAT&I))));
    %END;

    /**** CREATE OUTPUT SAS DATASETS OF THE COMBINED GENERALIZED INVERSES, ****/
    /**** THE DETERMINANTS OF EACH MATRIX AND THE RANKS OF EACH MATRIX *****/
    CREATE _FINMAT FROM ALLG;
    APPEND FROM ALLG;
    CREATE _FINDET FROM ALLD;
    APPEND FROM ALLD;
    CREATE _FINRANK FROM ALLR;
    APPEND FROM ALLR;
QUIT;

/**** CONVERT THE VALUES IN THE COMBINED GENERALIZED INVERSE, THE RANKS AND ****/
/**** THE DETERMINANTS DATASETS INTO ONE OBSERVATION PER COMBINATION OF BY ****/
/**** VARIABLE VALUES AND COMBINE THOSE DATASETS TOGETHER SO THEY CAN BE *****/
/**** MERGED TOGETHER WITH THE OTHER DATA TO USE THE VALUES IN THE GENERALIZED**/
/**** INVERSE DATASET AS THE WEIGHTS TO CALCULATE THE STUART-MAXWELL STATISTIC**/

DATA _GENINV(DROP=COL1-COL&DIM2R I _TCNT);
    SET _FINMAT;

    ARRAY GENINV (&DIMSQ2R) U1-U&DIMSQ2R;
    ARRAY COL (&DIM2R);

```

```

RETAIN U1-U&DIMSQR;

/**** INITIALIZE COUNTER VARIABLE ****/
IF _N_ = 1 THEN DO;
    _TCNT = 1;
    X = 1;
END;

/**** OBSERVATION WITH ALL MISSING VALUES IS THE ONE NOT USED FROM THE ****/
/**** ORIGINAL COVARIANCE MATRIX DUE TO THE DIMENSION REDUCTION *****/
/**** THIS INDICATES THE LAST OBSERVATION FOR A PARTICULAR BY-GROUP *****/
/**** COMBINATION AND WHERE THE FINAL WEIGHT VALUES SHOULD BE OUTPUT *****/
IF N(COL1-COL&DIM2R)=0 THEN DO;
    OUTPUT;
    X + 1;
    _TCNT = 1;
END;
ELSE DO I = 1 TO &DIM2R;
    GENINV(_TCNT) = COL(I);
    _TCNT + 1;
END;
RUN;

DATA _DET(DROP=COL1);
    SET _FINDET;
    X = _N_;
    DET = COL1;
RUN;
PROC SORT DATA=_DET;
    BY X;
RUN;

DATA _RANK(DROP=COL1);
    SET _FINRANK;
    X = _N_;
    RANK = COL1;
RUN;
PROC SORT DATA=_RANK;
    BY X;
RUN;

/**** KEEP ONLY ONE OBSERVATION PER COMBINATION OF BY-VARIABLES ****/
/**** (USED TO LABEL THE OBSERVATIONS FROM THE OTHER DATASETS) *****/
PROC SORT DATA=_STUNV2A(KEEP=&_BYVAR2 &_TREAT) OUT=_EXTRA;
    &_BYVAR &_TREAT;
RUN;
DATA _EXTRA;
    SET _EXTRA;
    &_BYVAR &_TREAT;
    IF FIRST.&_TREAT;
        X = _N_;
RUN;
PROC SORT DATA=_EXTRA;
    BY X;
RUN;

DATA _STUNV2B (DROP=X);
    MERGE _EXTRA _RANK _DET _GENINV;
    BY X;
RUN;

/***** COMBINE GENERALIZED INVERSE VALUES, RANKS AND DETERMINANTS *****/
/***** WITH ALL OTHER STUART-MAXWELL VALUES TO CALCULATE STATISTIC *****/
PROC SORT DATA=_STUNV2A;
    &_BYVAR &_TREAT;
RUN;

PROC SORT DATA=_STUNV2B;
    &_BYVAR &_TREAT;
RUN;

DATA _STUNV2C;
    MERGE _STUNV2A _STUNV2B;
    &_BYVAR &_TREAT;
RUN;
/***** END OF CODE ADDED BY SBLACK ON 22APR1997 *****/

```

```

DATA _STUINV2;
  LENGTH STATTYPE $25;
  SET _STUNV2C;

  ARRAY NN(&DIMPL2)          NN1-NN&DIMPL2 ;
  ARRAY D(&DIM2)              D1-D&DIM2    ;
  ARRAY V(&DIM2,&DIM2)         V1-V&DIMSQ2  ;
  ARRAY U(&DIM2R,&DIM2R)       U1-U&DIMSQ2R ;
  ARRAY A(&DIM2,&DIM2)         A1-A&DIMSQ2  ;

  /***** INVERT THE COVARIANCE MATRIX WITH THE SWEEP OPERATOR *****/
  /***** IF THE MATRIX IS SINGULAR, THE INVERSE VALUES WILL BE MISSING **/
  DO _K = 1 TO &DIM2R ;

    IF V(_K,_K) = 0 THEN _DD = . ;
    ELSE _DD = 1 / V(_K,_K) ;

    V(_K,_K) = 1 ;

    DO _I = 1 TO &DIM2 ;
      V(_K,_I) = _DD*V(_K,_I) ;
    END ;

    DO _J = 1 TO &DIM2 ;
      IF _J NE _K THEN V(_J,_K) = -V(_J,_K)*_DD ;
    END ;

    DO _J = 1 TO &DIM2 ;
      IF _J NE _K THEN DO ;
        DO _I = 1 TO &DIM2 ;
          IF _I NE _K THEN DO ;
            V(_J,_I) = V(_J,_I) + V(_J,_K)*V(_K,_I)/_DD ;
          END ;
        END ;
      END ;
    END ;

  END ;

  /*** CALCULATE CHI-SQUARE STATISTIC AND P-VALUE FOR THE TRUE STUART-MAXWELL STATISTIC ***/
  /*** AS WELL AS THE GENERALIZED VERSION OF THE STUART-MAXWELL STATISTIC *****/

  /*** _AVALUE IS THE TRUE STUART-MAXWELL STATISTIC *****/
  /*** _GVALUE IS THE GENERALIZED VERSION OF THE STUART-MAXWELL STATISTIC *****/
  DO _I = 1 TO &DIM2R ;
    DO _J = 1 TO &DIM2R ;
      _AVALUE = SUM( _AVALUE, V(_I,_J)*D(_I)*D(_J) ) ;
      _GVALUE = SUM( _GVALUE, U(_I,_J)*D(_I)*D(_J) ) ;
    END ;
  END ;

  _APVALUE = 1 - PROBCHI(_AVALUE,&DIM2R) ;
  IF RANK > 0 THEN _GPVALUE = 1 - PROBCHI(_GVALUE,RANK) ;
  ELSE _GPVALUE = . ;

  IF _APVALUE LE 0.05 AND _APVALUE NE . THEN _ASIGP='**' ;
  IF _GPVALUE LE 0.05 AND _GPVALUE GT 0 THEN _GSIGP='**' ;

  /*** ASSIGN APPROPRIATE VALUES TO THE STATISTICS TO BE PRINTED ***/
  IF _APVALUE = . AND _GPVALUE = . THEN DO;
    STATTYPE = 'NO STATISTIC CALCULATED';
    GENFLAG = .;
    _VALUE = .;
    _PVALUE = .;
    _SIGP = '';
  END;
  ELSE IF _APVALUE = . AND _GPVALUE ^= . THEN DO;
    STATTYPE = 'STUART-MAXWELL**';
    GENFLAG = 1;
    _VALUE = _GVALUE;
    _PVALUE = _GPVALUE;
    _SIGP = _GSIGP;
  END;
  ELSE IF _APVALUE ^= . THEN DO;
    STATTYPE = 'STUART-MAXWELL';
    GENFLAG = .;
    _VALUE = _AVALUE;
    _PVALUE = _APVALUE;
    _SIGP = _ASIGP;
  END;

```

```

RUN;

%END;    /** OFF-DIAGONAL VALUES PRESENT **/

%END ;   /** IF NEW > 2 **/

%ELSE %IF &NEW < 2 %THEN %DO;  /** IF NEW < 2 **/
    DATA _STUINV2;
        LENGTH STATTYPE $25;
        SET _STUINV1;
        STATTYPE = 'NO TEST POSSIBLE';
        GENFLAG = .;
        _VALUE = .;
        _PVALUE = .;
        _SIGP = ' ';
%END;    /** IF NEW < 2 **/

DATA _STUMAX;
    SET _STUMAX _STUINV2 ;
    IF TRIM(LEFT(&_TREAT)) ^= '';
    KEEP &_BYVAR2 &_TREAT STATTYPE GENFLAG _VALUE _PVALUE _SIGP;
RUN;

%END;    /** Inner Loop for NEW Values with Data **/

%END;    /** Master Loop for NEW Values **/

DATA &OTDSN;
    SET _STUMAX;
RUN;

/** Delete All Work Datasets Created by Macro **
*** Unless User Specifies Otherwise *****/
%IF &_DATDEL = YES %THEN %DO;
    PROC DATASETS;
        DELETE _STUMAX _STUFREQ _STUINV _STUINV1 _STUMAXI _STUNV2A _STUNV2B _STUNV2C
            _MATRX _PINMAT _FINDET _FINRANK _GENINV _DET _RANK _EXTRA _STUINV2;
    RUN;
%END;

*-----*
%MEND STUDAT01 ;

```