THESIS

This is to certify that the

thesis entitled

STRUCTURAL EQUATION MODELS
APPLIED TO HIERARCHICAL DATA

presented by

Joseph Michael Wisenbaker

has been accepted towards fulfillment
of the requirements for

Ph. D. degree in Counseling,
Personnel Services
and Educational
Psychology

William H. Schmidt
Major professor

Date 5/16/80

O-7639

STRUCTURAL EQUATION MODELS
APPLIED TO HIERARCHICAL DATA

by

Joseph Michael Wisenbaker

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Counseling, Personnel Services
and Educational Psychology

1980

ABSTRACT

STRUCTURAL EQUATION MODELS
APPLIED TO HIERARCHICAL DATA

by

Joseph Michael Wisenbaker

If one were to draw a sample of m classrooms and, within each classroom, observe n students on some set of variables of interest, one would obtain a variance-covariance matrix which could be decomposed into a simple linear combination of two other variance-covariance matrices; one arising at the classroom level and one at the subject's within-classroom level. Letting Σ_z represent the overall variance-covariance matrix, Σ_b the between-classroom variance-covariance matrix, and Σ the student's within-classroom variance-covariance matrix, we have

$$\Sigma_z = \Sigma_b + \Sigma.$$

While estimates of the within- and between-groups variance-covariance matrices may be of some interest in and of themselves, concern here is focused on a general parameterization of each, patterned after the structural analysis of variance-covariance matrices advocated by Jöreskog over the past decade.

Based on previous work by Schmidt, a technique for producing the maximum likelihood estimates for the parameters in the model is set forth. In addition, a chi-square test of fit of the model is presented along with an approach for producing the asymptotic variance-covariance matrix of the estimates from which asymptotic standard errors may be derived. Several sets of artificial data are analyzed using a computer program implementing this approach. Additionally, a real set of data drawn from the National Longitudinal Study of the High School Class of 1972 was used in an attempt to apply these techniques to a practical problem.

Difficulties in producing fully-converged estimates were noted with the analyses of the NLS data and one of the sets of artificial data. The author speculates on the factors contributing to the failure of the iterative techniques and suggests a strategy which may overcome this problem.

DEDICATION

This dissertation is dedicated to Cathy, who has been there since the beginning; and Michael, who came at the end.

ACKNOWLEDGEMENTS

As with all dissertations, none of this would have been possible without the help of too many people to specifically thank here. Those who were the most outstanding in their patience and encouragement include my advisor, Dr. William Schmidt; my good friend, Dr. John Schweitzer; and my wife, who endured it all.

I would also like to express my gratitude to Research Triangle Institute, which has offered a great deal of moral and concrete support for my research since hiring me two and one-half years ago. Deserving special mention are Dr. Junius A. Davis, the Director of the Center for Educational Research and Evaluation; and Barbara Elliott, who spent innumerable hours in preparing the many drafts and the final manuscript.

TABLE OF CONTENTS

	Page
List of Tables	vii
List of Figures	viii
Chapter 1	
Introduction and Statement of the Problem	1
The General Covariance Structure Model	5
Chapter 2	
Literature Review--Preliminary Development	7
Jöreskog's Contributions to Structural Equation Modeling	8
Some Contributions by the Chicago Group	16
Chapter 3	
Schmidt's Hierarchical Model	19
Jöreskog's Linear Structural Equation System Model	24
Chapter 4	
General Structural Equation Model for Hierarchical Data	28
Parameter Estimation: General Considerations	31
Numerical Solutions for Parameter Estimates	32
Steepest Descent	34
Davidon-Fletcher-Powell	35
Identifiability	36
Estimating Parameters in the General Model	39
Standard Error Estimation and Test of Fit of the Estimated Model	42

TABLE OF CONTENTS (continued)

Page

Chapter 5

Applications 46

Analysis of Artificial Data: Testing the Estimation Procedure Using a Simple Model 46

Analysis of Artificial Data: Testing the Estimation Procedure Using a Complex Model 59

Analysis of Data Drawn from the National Longitudinal Study of the High School Class of 1972 65

Analysis of a Final Set of Artificial Data 69

Attempted Solutions for the Estimation Problem 73

An Illustrative Interpretation of the NLS Results 78

Chapter 6

Summary of Results, Conclusions, and New Directions 86

List of References 96

Appendix A

First Derivative of the Log Likelihood Function with Respect to the Parameter Matrices A1

Appendix B

First Derivative of Σ with Respect to Individual Elements of the Parameter Matrices B1

Appendix C

Second Derivative of Σ with Respect to Individual Elements of the Parameter Matrices C1

TABLE OF CONTENTS (continued)

Page

Appendix D

Deck Setup for Use of Standard Error Routine D1

Appendix E

Listing of Standard Error Program E1

Appendix F

Listing of Estimation Program F1

LIST OF TABLES

Table		Page
1	Parameterizations Employed in Analyzing Artificial Data from Schmidt	49
2	Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 1	50
3	Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 2	53
4	Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 3	55
5	Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 4	57
6	Estimated Values for Parameters Treated as Not Fixed in Example II	64
7	Intermediate Parameter Estimates for NLS Data	80

LIST OF FIGURES

Figures		Page
1	Artificial Data Obtained from Schmidt (1969)	47
2	Parameter Values Used to Generate Variance-Covariance Matrices for Example II	61
3	Values of Σ and Σ_b for Artificial Data Example II . . .	62
4	Values of S and S_b for Artificial Data Example II . . .	63
5	Lower Triangular Elements of the Observed Within-School Variance-Covariance Matrix from NLS Data	67
6	Lower Triangular Elements of the Observed Between-School Variance-Coveriance Matrix from NLS Data . . .	68
7	General Diagrammatic Structure of Estimated Model . . .	70
8	General Parameterization of Model Variance-Covariance Matrices	71
9	Parameter Values Used to Generate Variance-Covariance Matrices for Example IV	72
10	Values of Σ and Σ_b Matrices for Example IV	74
11	Lower Triangular Elements of the S and S_b Matrices for Example IV	75
12	General Model Underlying Example IV	76
13	General Diagrammatic Structure of Estimated Model Using NLS Data With Selected Parameter Estimates . .	82

Chapter 1

Introduction and Statement of the Problem

As Kerlinger has pointed out, the history of the sciences has as its unifying thread the search for relationships among variables. Viewed in the light of this underlying activity, progress in scientific methodology involves finer and finer refinements in our ability to search out and evaluate the nature of relationships among variables. While the theoretical organization of our knowledge about various interrelationships may indeed be subject to progress via revolution à la Kuhn (1970), progress in statistical methodology can more reasonably be viewed as evolutionary in the sense that our later approaches allow us to model relationships subject to fewer unrealistic constraints than those approaches previously available.

At a very fundamental level, we may try to assess the nature of the relationship between two observed variables. One candidate for this task is the correlation coefficient, which estimates the strength of the linear association between two variables. So long as we wish to deal with a situation where there are but two variables, where the variables are not conceptualized as dependent and independent, and where our measurement processes are assumed to be error free, there is no real drawback to this approach.

If, on the other hand, we admit the consideration of a slightly more complex model where one variable is seen as dependent on another, a different approach is called for. The logical method to turn to in this instance is regression analysis, whereby we can estimate the magnitude of one variable given various values of the other. Once again, this involves a relatively simple model of the real world. The fact is that often there are many variables, each related to varying degrees with others, which are related to a particular dependent variable. To adequately investigate the simultaneous impact of each independent variable, we may use the multiple regression approach. Examination of the regression coefficients associated with the independent variables allows us to determine the nature of the conditional relationship between each independent variable and our dependent variable.

Allow us to jump ahead several magnitudes in the complexity of the model we are willing to consider. If we allow not only for multiple independent variables but also for some of them to "causally depend" upon others and in turn "cause" still others, we have quite a complex model indeed. If we are dealing with what has been termed a "recursive" model, estimates of the coefficients which must simultaneously hold can be done through the repeated application of regression analysis with each presumed "causally dependent" variable functioning as the dependent variable for one analysis and as an independent variable in subsequent analyses. A great deal of the more sophisticated work in the sociological literature has, of late, fallen into this category (Miller, et al. [1979], Mortimer and Lorence [1979], Kohn and Schooler [1978], Bielby, Hauser, and Featherman [1977]).

In those instances where models are non-recursive (i.e., where variables may simultaneously affect each other) more complex estimation procedures are called for. Two and three stage generalized least squares represent one approach; maximum likelihood estimation another.

A still greater step toward dealing with more "realistic" models is to take into account the measurement error associated with the variables involved and the fact that a number of the measures may, in fact, be addressing the same latent trait. Jöreskog has recently formulated a general mathematical statement of such a model and, with Sörbom, generated a computer program to provide estimates of the various parameters via maximum likelihood.

At this point, one may well ask why further refinements are needed. In fact, there are applications in many fields for models of such complexity. Unfortunately, there is an issue which, while not completely unique to educational problems, presents further complexity still.

While the educational psychologist operating as a "pure" psychologist may explore aspects of learning theory in the relatively safe confines of the experimental laboratory, the educational psychologist operating in the context of the classroom is faced with a variety of problems. Often he lacks the authority to carry out the random assignments of subjects to various conditions in which he is interested. He has no choice but to rely upon correlational approaches in his search for knowledge, thereby making the utilization of such models as those formulated by Jöreskog a logical approach. But that same lack of control which seems to point toward the use of sophisticated structural equation models carries with it another problem--students do not receive their instruction individually but in groups. What should be our unit of analysis? Ought we to ignore the inherent hierarchical structure in our data sets in favor of a simplistic approach? Or should the educational psychologist recast his thinking in terms of sociology and explore his questions in terms of classrooms rather than individuals? Neither approach has intuitive appeal. In the first instance, we confound classroom-level effects with individual-level effects; in the second, we lose the ability to unambiguously apply that knowledge gained under laboratory conditions.

A variety of papers dealing with the analysis of multi-level data have appeared in recent sessions of the Annual Meeting of AERA. Motivated in part by Leigh Burstein's interest in this area, these paper sessions have provided a forum for a variety of discussions. Several papers dealt with the problems associated with the analysis of data aggregated to a higher level (Maw [1976], Hannan [1976]). Others have considered the choice of the appropriate unit of analysis in the context of ANOVA (Glendening [1976]). Burstein (1976) came closest to the spirit of the present undertaking when he recommended that analysis be carried out at the lowest level at which observations could be considered to be independent. His more recent work (Burstein [1979]) has tended to focus on the use of within-groups regression coefficients broadly examined at the between-groups level.

Perhaps the most logical answer to the difficulty of analyzing data arising from two levels is to try to assess the nature of intervariable relationships at both levels simultaneously. Cronbach (undated) has come to advocate such an approach through the analysis of both within- and between-groups variance-covariance matrices. Thus far, his efforts have revolved around the use of regression analysis at the subjects within-groups and the between-groups levels respectively. The most likely extension of this approach is in the direction of using structural equation models rather than regression models.

There are difficulties, however, in such an approach. In a slightly different context, Schmidt (1969) has pointed out that, conceptually speaking, the between-groups variance-covariance matrix has, as its expected value, components due to both individual and group levels. Simple adjustment of the between-groups variance-covariance matrix is precluded by the non-zero probability of negative variance estimates. Schmidt has developed a maximum likelihood approach to the estimation of the two variance-covariance matrices which overcomes this problem and has elaborated upon it in the context of the analysis of covariance structures.

The next logical step is to apply Schmidt's approach to the analysis of hierarchical data to the estimation of structural equation models patterned after Jöreskog. That is the task which the present author has undertaken, and reports herein.

The General Covariance Structure Model

Since a few general conditions underlie all of the work discussed in the following pages, it may be best to set them forth at this point to provide a common focus for subsequent discussion. The general covariance structure model has, as its basis, the following fundamental equation linking the observed multivariate "outcome" vector, $\underline{y}$, to a similarly multivariate "causal" vector, $\underline{\theta}$:

$$\underline{y} = \Lambda \underline{\theta} + \underline{e}$$

In this situation, Λ is a matrix of coefficients relating the elements of $\underline{\theta}$ to those of $\underline{y}$, and $\underline{e}$ is a vector of errors associated with that relationship.

Under the assumption that the covariance between $\underline{\theta}$ and $\underline{e}$ is identically equivalent to a matrix of zeroes and with the definitions

$$E(\underline{e}) = 0,$$

$$V(\underline{e}) = \psi,$$

and

$$V(\underline{\theta}) = \Phi$$

where both ψ and Φ are square matrices of the appropriate dimensions, two general implications obtain. In mathematical notation, these are:

$$E(\underline{y}) = \Lambda \, E(\underline{\theta})$$

and

$$V(\underline{y}) = \Lambda \, \Phi \, \Lambda' + \psi.$$

The first of these simply states that the expected value of the vector $\underline{y}$ is a simple linear combination of the expected values of the vector $\underline{\theta}$. In subsequent discussions, this fact is generally disregarded by the simplifying supposition that the expected value of $\underline{\theta}$ is equal to a vector of zeroes. Far more important is the second implication, which states that the variance of $\underline{y}$ is a linear combination of the variance of $\underline{\theta}$ plus the variance associated with the error term $\underline{e}$. Since all of the various parameters normally of interest in the underlying model are reflected in this relationship, it is just this relationship that has been the focus of a great deal of statistical thought and elaboration.

Chapter 2

Literature Review--Preliminary Developments

To appreciate the development of the statistical approach to structural equation modeling, we must refer back to the work of Lawley, who provided a statistical basis for the estimation and testing of factor analytic models. Prior to Lawley's work, the classical methods of factor analysis were based upon algebraic transformations of the correlation matrix without regard for sampling theory or statistical tests of fit. Harman (1967) cites the work carried out by Lawley in the early 1940's as the first to attack the problem of estimating factor loading matrices from a statistical standpoint. Employing the multivariate normal distribution derived by Wishart (1928), Lawley (1940) produced the partial derivatives of the logarithm of the likelihood function with respect to each of the elements in the factor analytic model:

$$\Sigma = \Lambda \, \Phi \, \Lambda' + \Psi. \qquad (1)$$

In this model, Σ represents the "true" variance-covariance matrix of the variables involved, Λ is a matrix of factor loadings, and ψ corresponds to a diagonal matrix containing the unique variance associated with each variable. The factors are assumed implicitly to be orthogonal with unit variance ($\Phi = I$).

When the partial derivatives of the log likelihood function with respect to Λ and ψ are set equal to zero, the simultaneous solution of the resulting system of equations will yield the maximum likelihood estimates for Λ and ψ. Because of the complexity of this sytem of equations, analytic solutions for the parameter estimates can be obtained only in very simple cases. Thus the general application of this approach must rely on numerical methods for solving the system of equations. According to Harman (1967), the computational burden imposed by the iterative method advanced by Lawley discouraged the use of maximum likelihood factor analysis in the 1940's and 1950's.

Harman (1967) notes that several investigators in the early 1960's, including Harman, Hemmerle and Jöreskog carried out work aimed at producing efficient, computer-based procedures for arriving at iterative solutions for the parameter estimates. It is just such work that has made feasible the use of maximum likelihood factor analysis.

While many researchers have made important contributions toward the development of the statistical approach to structural equation modeling, the complexity of the issue argues for some unified approach to examining the sequence of developments. The approach adopted here is to follow the work of Jöreskog. While his work is, in many respects, not unique nor even the most pioneering in many instances, it does represent the single most sustained effort toward the development and implementation of an approach to structural equation modeling available in the literature.

Jöreskog's Contributions to Structural Equation Modeling

As indicated by Harman (1967), Jöreskog's earliest work in this area was directed at implementing the maximum likelihood approach to factor analysis originated by Lawley. In Jöreskog's first journal article dealing with maximum likelihood factor analysis (1966), he discussed an approach whereby a simple structure hypothesis might be tested. Being essentially an operationalization of Thurstone's notion of simple structure in the context of maximum likelihood factor analysis, the article began by positing a factor analytic model for the variance-covariance matrix of a vector of random variables identical in form to equation (1). This is fundamentally the same model considered by Lawley with the relaxation of the constraint that the factors be uncorrelated and have unit variance. As in the work by Lawley, the variables involved were assumed to be multivariate normally distributed. Under this assumption, the logarithm of the likelihood function takes on the following form:

$$L = -\frac{1}{2} n \left[\log |\Sigma| + \mathrm{tr}\,(S\Sigma^{-1})\right]. \qquad (2)$$

In this equation, n is equal to the number of observations, Σ is the true variance-covariance matrix, and S is the observed variance-covariance matrix. It is essentially this likelihood function that provides the basis for the vast majority of the work in this area and virtually all of that carried out by Jöreskog.

In the article currently under discussion, Jöreskog was interested in testing the fit of the model in the situation where certain elements in the Λ matrix were constrained to be equal to zero, this being the criterion for the existence of simple structure. To carry out such a test, maximum likelihood estimates for the parameters not so constrained had to be obtained. Following Lawley, Jöreskog generated the derivatives of the log likelihood function with respect to the free elements of Λ, Φ, and Ψ. The resulting expressions were equated to zero and a numerical solution for the simultaneous set of resulting equations was sought.

In considering ways to generate numerical estimates, Jöreskog experimented with three different approaches. Those considered included the original approach described by Lawley (1958), the method of steepest descent, and the method of resultant descents. Based on their performance in analyzing several sets of data, Jöreskog argued for the latter approach as the most efficient method of the three. Jöreskog implemented this method in a computer program to produce maximum likelihood estimates for the factor analytic model permitting elements in Λ^* to be constrained equal to zero.

The basis for testing the fit of the model was the following likelihood ratio:

$$\chi^2 = -\frac{1}{2} n \left[\log |\hat{\Sigma}| - \log |S| + \text{tr} \{S\hat{\Sigma}^{-1}\} - p \right] \tag{3}$$

where n and S are defined as before, $\hat{\Sigma}$ is the estimate of Σ determined by $\hat{\Lambda}$, $\hat{\Phi}$, and $\hat{\Psi}$, and p is the number of variables in Σ. Under the null hypothesis that the particular model involved fits the data and for reasonably large n, the statistic is distributed approximately chi-square with the degrees of freedom equal to the number of independent elements in S less the number of parameters estimated in the model. This approach to testing the fit of a model estimated via maximum likelihood is the one uniformly adopted in all work considered herein and is another constant found throughout Jöreskog's work.

Whereas the situation considered in the previous article was essentially a simple version of confirmatory factor analysis, Jöreskog's next article (1967) dealt with the implementation of Lawley's exploratory factor analytic model. To ensure the identifiability of the model, the variance-covariance matrix of the factors was, once more, constrained equal to an identity matrix. This yields the same basic model considered earlier and expressed in equation (1) with Φ, once again, constrained equal to an identity matrix.

At this point, Jöreskog made two contributions which assisted in the popularization of maximum likelihood factor analysis. The first had to do with the implementation of a new approach to solving iteratively for parameter estimates, while the second addressed the problem of inadmissable solutions.

Based on his experience with the use of the method of resultant descents, Jöreskog expressed dissatisfaction with its rate of convergence in some instances. As an alternative, he adopted the Fletcher-Powell (1963) method, with which he experienced general success. This method provided the basis for the estimation approach used throughout his more recent work.

With respect to the problem of inadmissible solutions, Jöreskog indicated that, with the estimation procedures heretofore employed, there had been no guarantee that one or more of the elements in Ψ could not become negative. Not only is such a situation inadmissible because Ψ is supposed to be a matrix containing only variances in the diagonal, but the attainment of such values, according to Jöreskog, frequently heralds the complete breakdown of the estimation procedure. Since the Fletcher-Powell method proved to be just as susceptible to this problem as the ones previously tried, Jöreskog imposed the restriction that none of the elements in Ψ could be less than some arbitrarily small positive value. This method was implemented in a computer program called UMLFA (1966) and made generally available.

In Jöreskog (1969) we have the final developments in what can be considered a purely factor analytic model. Subsequent articles, while adopting the same strategy toward estimating parameters and testing the fit of the estimated model, dealt with models of a more general nature. Expanding on an earlier article by Jöreskog and Lawley (1968), Jöreskog dealt not merely with confirmatory maximum likelihood factor analysis, as in the 1966 article, nor with the exploratory version as in 1967; rather, interest was placed on a more general approach designated as restricted maximum likelihood factor analysis. In this model, parameters could be of two kinds, fixed or free. The free parameters were those to be estimated from the data at hand; the fixed parameters were assumed equal to certain fixed values. The flexibility here is that some or all of the elements in any one or more of the parameter matrices may be fixed equal to any chosen constants. Given these options, Jöreskog returned to the more general formulation of the factor analytic model expressed in equation (1). With appropriate restrictions on the elements of the various parameter matrices, parameters in any or all of the parameter matrices may be estimated. From this point on, model identifiability can be addressed through the use of more specific and research-based a priori restrictions than the expedient of constraining Φ to be equal to an identity matrix.

Since it was known at this time that the Fletcher-Powell method might not converge if its starting values were too far from the correct values, Jöreskog reported an additional modification to his estimation routine. Rather than simply starting off with the Fletcher-Powell method, the first stage of the iterative solution incorporated a number of steepest descent iterations to obtain a better starting point for Fletcher-Powell. This too is a characteristic of Jöreskog's remaining work.

In Jöreskog (1970) we find a more general model than those previously considered. This model has added several parameter matrices for modeling Σ and, in addition, sets forth a model for the expected value of the variables involved. The model has the following form:

$$\Sigma = B(\Lambda \; \Phi \; \Lambda' + \Psi^2)B' + \theta^2$$
$$E(X) = A \quad \Xi \quad P \tag{4}$$

where X is an nxp observational matrix, A and P are fixed matrices with dimensions nxg and hxp respectively, Φ is symmetric, Ψ^2 and θ^2 are diagonal, and B and Λ are rectangular. The matrix Ξ is a matrix of latent values, while A and P serve to reparameterize these to the matrix of observed values. The addition of B and θ^2 serve to make this a second-order factor analytic model. This model permitted parameterizations of both the means and variance-covariance matrices. In addition to the constraints on the elements permitted in the previous article, Jöreskog introduced a third and final class of restrictions--elements of the parameter matrices could be constrained equal to one another but estimable otherwise. This additional item of flexibility completed the list of options available in Jöreskog's work hereafter.

With the addition of the parameterizations permitted on the matrix of expected values, the log likelihood function has as its more complete form:

$$\log L = -\frac{1}{2} pn \log (2\pi) - \frac{1}{2} n \log |\Sigma|$$
$$- \frac{1}{2} \sum_{\alpha=1}^{n} \sum_{i=1}^{p} \sum_{j=1}^{p} (X_{\alpha i} - \mu_{\alpha i}) \, \sigma_{ij} \, (X_{\alpha j} - \mu_{\alpha j}) \tag{5}$$

where $\mu_{\alpha i}$ is an element of E(X) and σ_{ij} of Σ^{-1}.

Jöreskog also formalized the process of maximizing log L given the possible restrictions on the parameters. To do so, he considered the elements of the parameter matrices to be arranged as a vector, $\underline{z}$, containing k elements. As a result, the logarithm of the likelihood function became a function of $\underline{z}$. If we designate $\partial F/\partial \underline{z}$ and $\partial^2 F/\partial \underline{z}\partial \underline{z}'$ as the first and second order derivatives respectively, fixed elements could be dealt with by assuming the first ℓ elements to be free, with the remaining $k-\ell$ fixed. This yields a function of only ℓ elements which can be designated as a vector $\underline{y}$ where the first and second derivatives can be designated as $\partial G/\partial \underline{y}$ and $\partial^2 G/\partial \underline{y}\partial \underline{y}'$ respectively. These may be obtained from $\partial F/\partial \underline{z}$ and $\partial^2 F/\partial \underline{z}\partial \underline{z}'$ by omitting appropriate rows and columns in these matrices.

Assuming the existence of but m distinct parameters designated as $\underline{x}$, the issue of constraining some parameters to be equal to others was handled by defining elements of a matrix M as follows:

$$M_{ig} = \begin{cases} 1 \text{ if } y_i = x_g \\ 0 \text{ otherwise.} \end{cases} \tag{6}$$

The logarithm of the likelihood function was now expressed as a function $H(\underline{x})$ where

$$\partial H/\partial \underline{x}_g = \sum_{i=1}^{\ell} \partial G/\partial \underline{y}_i \, M_{ig} \tag{7}$$

and

$$\partial^2 H/\partial x_g \, \partial x_h = \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \{\partial^2 G/(\partial y_i \partial y_j)\} \, M_{ig} \, M_{jh} \, . \tag{8}$$

Thus, the logarithm of the likelihood function was to be maximized by applying the Fletcher-Powell method to solving the simultaneous equations resulting from setting $\partial H/\partial \underline{x}$ to zero.

Jöreskog then illustrated the application of this general model in the contexts of congeneric measurements as developed by Lord and Novick (1968), factor analysis, variance-components estimation as set forth by Bock and Bargmann (1966), analysis of ordered response following Pothoff and Ray (1964), and path analysis following Wright (1918).

In another article during the same year, Jöreskog (1970) dealt with the same model discussed above. No new theoretical or procedural results were introduced, but the application of the model to the estimation of parameters associated with the Werner Simplex and Quasi-Simplex were discussed and illustrated. In an article in the same vein, Jöreskog (1971) illustrated the application of this model to estimating parameters in models dealing with congeneric tests. Again in 1973, Jöreskog employed the same model to estimate parameters in test theory models, congeneric tests, multitrait-multimethod data, factor analysis, variance-covariance components, simplex and circumplex models, and path analytic models. The same approach and types of applications are also found in Jöreskog (1974).

Jöreskog's most recent embellishments on this basic model (1975, 1977) permitted explicitly dealing with situations in structural equation modeling characterized by the presence of both endogenous and exogenous variables measured with error. Since it is the purpose of the work reported herein to extend this class of models to the situation in which hierarchical data is to be analyzed, this model is more fully discussed in the following chapter.

Some Contributions by the Chicago Group

As has been mentioned previously, the types of models considered by Jöreskog were not unique to his work. Maximum likelihood factor analysis models were dealt with independently by a number of workers in the area including Hemmerle (1965) and Harman (1966).

A maximum likelihood approach to estimating the parameters in a model virtually identical to that discussed by Jöreskog (1975) was discussed by Wiley (1973) at the same paper session at the University of Wisconsin at Madison in 1970. A line of inquiry represented by the work of Bock (1960); Bock and Bargmann (1966); Wiley (1967); and Wiley, Schmidt, and Bramble (1973) addressed a set of models formally parameterized as the factor analysis model but with different notions as to the roles of the parameters themselves. It is this set of papers that led to the work of Schmidt (1969) which extended the application of similar models to the situation where observations were nested within higher order units. It is to this line of inquiry that we now turn.

Bock (1960) argued that the similarity noted by Burt (1947) and Creasy (1957) between factor analysis and analysis of variance can, under the proper circumstances, be considered a "formal relation." He further stated that if tests are chosen based on specific hypotheses relative to their composition, a mixed model analysis of variance could be used to examine their structural and distributional properties. The purpose of using the mixed model ANOVA was to avoid the statistical problems inherent in factor analysis at that time. Operationally, the approach advocated by Bock merely permitted (at least conceptually) an investigator to effectively fix the Λ matrix in the factor analytic model equal to the design matrix associated with the tests. The statistical problems normally associated with classical factor analysis could then be dealt with in the framework of ANOVA.

Bock and Bargmann (1966) essentially reversed this line of argument with their discussion of the analysis of covariance structures operationalized through maximum likelihood estimation procedures. The fundamental model they addressed treated data arising from a random sample of subjects for whom observations were assumed to be multivariate normally distributed with some arbitrary mean $\underline{\mu}$ and a variance-covariance matrix with the following structure:

$$\Sigma = \Lambda \, \Phi \, \Lambda' + \Psi \tag{9}$$

where Λ is a matrix of known coefficients of the linear functions connecting the observed and latent variables, Φ is the variance-covariance matrix of latent variables, and Ψ is the diagonal matrix of measurement error variances.

They considered the estimation of Φ and Ψ under three conditions. The first was where Φ was constrained to be diagonal and the diagonal elements of Ψ were equal. In the second, the diagonal elements of Ψ were allowed to be unequal. Finally, the third condition specified an additional parameter matrix of scale factors in the diagonals such that

$$\Sigma = \beta(\Lambda \, \Phi \, \Lambda' + \Psi)\beta' \tag{10}$$

where the diagonal elements of Ψ were, once again, constrained to be equal. Bock and Bargmann went on to demonstrate the derivation of the first derivatives of the likelihood function with respect to each of the parameter matrices in the three models, set forth the appropriate likelihood ratio tests of fit, and discussed the iterative scheme for parameter estimation employing the Newton-Raphson method.

While the form of these models is hauntingly similar to that of Jöreskog (1970), their development would appear to have taken place independently, for nowhere in their article is Lawley's work on maximum likelihood factor analysis cited. Furthermore, their conceptualization of the applicability of such models appears to foreshadow applications later made by Jöreskog (1971).

Wiley, Schmidt, and Bramble (1973) later expanded on their work by considering 8 variations on the model defined by fully crossing the following conditions:

1) β is a general diagonal matrix of scale factors or an identity matrix,
2) Φ is diagonal or simply symmetric positive definite matrix, and
3) Ψ is diagonal with equal diagonal elements or the diagonal elements are allowed to differ.

Since this paper was based on a somewhat earlier paper by Bramble, Schmidt, and Wiley, some of the conditions considered were also dealt with by Schmidt (1969).

Chapter 3

Schmidt's Hierarchical Model

In his doctoral dissertation in 1969, Schmidt set out to implement one of the covariance structure models then being developed simultaneously at the University of Chicago by Wiley and others, and at the Educational Testing Service by Jöreskog. The major difference between his work and that of the others lay in the fact that his model was developed in the context of data arising from observations nested within groups. Owing to this, Schmidt's problem was that of setting out a way to simultaneously estimate two models, one at the within-groups level and the other at the between-groups level.

While it must be understood that Schmidt's model, like all covariance structure models, is most fundamentally derived from models of the underlying observations, previous discussion of the work in structural equation models provides sufficient grounding in this basic principle to allow us to proceed directly to the models for the variance-covariance matrices themselves.

First, the overall variance-covariance matrix Σ was seen as a simple additive function of the within- and between-groups variance-covariance matrices, Σ_w and Σ_b respectively. Their relationship was expressed as follows:

$$\Sigma = \Sigma_w + \Sigma_b. \tag{11}$$

Each of the two matrices, Σ_w and Σ_b, was expressed as a function of matrices relating observed to latent variables, variance-covariance matrices among the latent variables and, finally, variance-covariance matrices among the errors of measurement. These various matrices were assembled into one two-part model:

$$\Sigma_w = \Lambda_w \Phi_w \Lambda_w' + \Psi_w \tag{12}$$

$$\Sigma_b = \Lambda_b \Phi_b \Lambda_b' + \Psi_b \tag{13}$$

where

Λ_b and Λ_w represent the matrices of coefficients relating observed to latent variables at the between- and within-groups levels, respectively,

Φ_b and Φ_w represent the variance-covariance matrices among the latent variables at the between- and within-groups levels,

Ψ_b and Ψ_w represent the variance-covariance matrices among the errors of measurement at the between- and within-groups levels.

In general, Schmidt's model treated the matrices Ψ_w and Ψ_b as having non-zero values on the diagonals and zero elsewhere. In addition, Λ_w and Λ_b were permitted to be matrices containing known constants (as arising from an experimental design over the measures) or free parameters to be estimated. Finally, the Φ_w and Φ_b were allowed to be considered as diagonal matrices or general matrices with non-zero elements in the off diagonals as well.

Taking Σ_w and Σ_b separately, it can be readily noted that each model corresponds to those considered by Jöreskog (1967). The primary distinction is the fact that the models, each of which represents a set of simultaneous equations, are themselves intended as being simultaneously operative. Thus, the procedure adopted for estimating the parameters in a particular application must be capable of estimating parameters at both levels simultaneously. Schmidt's method of choice for this was the method of maximum likelihood, and key to its application was his formulation of the likelihood function for hierarchical data.

He began with a basic situation in which p measures were available for each of n subjects within each of m groups. Based on the work of Tiao and Tan (1965) he reconceptualized this situation as one in which each of the m groups were composed of np observations. Thus the data was treated as m independent observations drawn from an np dimensional multivariate normal distribution. This distribution had a mean of $\underline{1} \times \underline{\mu}$ and a covariance matrix, Σ_{np}, with the following structure:

$$\Sigma_{np} = \underline{1}\,\underline{1}' \otimes \Sigma_b + I \otimes \Sigma_\omega. \tag{14}$$

In this equation, Σ_ω represents the covariance between observations within each group while Σ_b represents the between-groups covariance.

Given the assumption of a multivariate normal distribution, Schmidt derived the following as an expression for the likelihood function:

$$L = (2\pi)^{\frac{-mnp}{2}} \; \Sigma_{np}^{\frac{-m}{2}} \; e^{\{-\frac{1}{2}[\sum_{i=1}^{m}((\underline{y}_i - \underline{1} \times \underline{\mu})' \Sigma_{np}(\underline{y}_i - \underline{1} \times \underline{\mu}))]\}} \tag{15}$$

Substituting the previously-defined expression for Σ_{np} in terms of Σ_ω and Σ_b into the above expression and simplifying making use of several matrix algebra theorems, Schmidt obtained the following as an expression for the likelihood function in terms of Σ_b and Σ_ω rather than Σ_{np}:

$$L = (2\pi)^{\frac{-mnp}{2}} \quad \Sigma_w^{\frac{m-mn}{2}} \quad (\Sigma_w + n\Sigma_b)^{\frac{-m}{2}}$$

$$e^{\{ -\frac{1}{2} [mn \text{ tr } \{\Sigma_w^{-1} S_w\} + m \text{ tr } \{(\Sigma_w + n\,\Sigma_b)^{-1} S_b\}} \tag{16}$$

$$+ mn \text{ tr } \{(\Sigma_w + n\,\Sigma_b)^{-1}(\bar{\underline{y}} - \underline{\mu})(\bar{\underline{y}} - \underline{\mu})'\}]\}$$

where

$$S_w = \frac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} (\underline{y}_{ij} - \underline{y}_{i..})(\underline{y}_{ij} - y_{i..})' \tag{17}$$

$$S_b = \frac{n}{m} \sum_{i=1}^{m} (\underline{y}_{i.} - \underline{y}_{..})(\underline{y}_{i.} - \underline{y}_{..})' \tag{18}$$

and

$\underline{y}_{ij}$ is a vector of length p for the $j^{\underline{th}}$ subject in the $i^{\underline{th}}$ group. The values of $\hat{\Sigma}_b$ and $\hat{\Sigma}_w$ which cause this function to attain its maximum are the maximum likelihood estimates for Σ_b and Σ_w. Since one of the properties of this form of estimation is that the same estimates for Σ_b and Σ_w will be obtained through maximizing any monotonic function of L, the usual approach is to maximize a slightly simpler function of L, namely the logarithmic function.

Schmidt derived the following as an expression for the logarithm of the likelihood function:

$$\log L = \frac{-mnp}{2} \log(2\pi) + \frac{m-mn}{2} \log\,(|\Sigma_w|) - \frac{m}{2} \log(|\Sigma_w + n\Sigma_b|)$$

$$- \frac{1}{2} [mn \text{ tr } \{\Sigma_w^{-1} S_w\} + m \text{ tr } \{(\Sigma_w + n\Sigma_b)^{-1} S_b\} \tag{19}$$

$$+ mn \text{ tr } \{(\Sigma_w + n\Sigma_b)^{-1} (\underline{y}_{..} - \underline{\mu})(\underline{y}_{..} - \underline{\mu})'\}].$$

Since the maximum likelihood estimator of $\underline{\mu}$ is $\underline{y}$, the last term in the above expression is zero. The first term being constant, it can be effectively ignored with no impact on any results. This yields the following as Schmidt's expression for the effective part of the log likelihood function:

$$\log L = \frac{m-mn}{2} \log (|\Sigma_w|) - \frac{m}{2} \log (|\Sigma_w + n\Sigma_b|) - \frac{mn}{2} \operatorname{tr} \{\Sigma_w^{-1} S_w\} - \frac{m}{2} \operatorname{tr} \{(\Sigma_w + n\Sigma_b)^{-1} S_b\}. \tag{20}$$

When a particular parameterization of Σ_b and Σ_w is substituted in this expression, maximum likelihood estimates for the parameters can be obtained by setting the first partial derivatives of log L with respect to each parameter equal to zero and solving for the unknown of interest. Because the approach adopted by the present author makes use of the chain rules for obtaining these derivatives, the first partial derivative of log L with respect to Σ_b and Σ_w are necessary. Schmidt's expressions for these are set out below:

$$\begin{aligned}\frac{\partial \log L}{\partial \Sigma} = {} & m(1-n)\Sigma_w^{-1} - m(\Sigma_w + n\,\Sigma_b)^{-1} + mn\,\Sigma_w^{-1} S_w \Sigma_w^{-1} \\ & + (\Sigma_w + n\,\Sigma_b)^{-1} S_b (\Sigma_w + n\Sigma_b)^{-1} \\ & - .5 \operatorname{diag}\{m(1-n)\Sigma_w^{-1} - m(\Sigma_w + n\,\Sigma_b)^{-1} + mn\,\Sigma_w^{-1} S_w \Sigma_w^{-1} \\ & + (\Sigma_w + n\,\Sigma_b)^{-1} S_b (\Sigma_w + n\,\Sigma_b)^{-1}\}\end{aligned} \tag{21}$$

$$\frac{\partial \log L}{\partial \Sigma_a} = mn\ (\Sigma_w + n\ \Sigma_b)^{-1}\ S_b\ (\Sigma_w + n\ \Sigma_b)^{-1} - mn\ (\Sigma_w + n\ \Sigma_b)^{-1}$$

$$- .5\ \mathrm{diag}\{mn\ (\Sigma_w + n\ \Sigma_b)^{-1}\ S_b\ (\Sigma_w + n\ \Sigma_b)^{-1} - mn\ (\Sigma_w + n\ \Sigma_b)^{-1}\}. \quad (22)$$

Jöreskog's Linear Structural Equation System Model

Two of the distinctive features of Jöreskog's most recent structural equation model are, first, that the structural relationships may be expressed in terms of latent variables and; second, that such variables are allowed to be fallibly measured. This implies that the overall model is expressible as two components, a measurement model and a structural model. The following presentation of Jöreskog's formulation is based on the ideas set forth in Jöreskog and Van Thillo (1972), Jöreskog (1973) and Jöreskog (1977) with notational modifications allowing for a ready comparison of Jöreskog's model with its extension to the hierarchical data situation to be presented subsequently.

Measurement Model

$$\underline{y} = \underline{\mu} + \Lambda\underline{\eta} + \underline{\varepsilon} \quad (23)$$

$$\underline{x} = \underline{v} + \Gamma\underline{\zeta} + \underline{w} \quad (24)$$

In this component of the overall model, the vector of y's represents the set of observed endogenous measures which have as their expected value $\underline{\mu}$ and error $\underline{\varepsilon}$. The vector of η's stands for the latent or "true" endogenous variables while Λ is a coefficient matrix relating $\underline{\eta}$ to $\underline{y}$. Likewise $\underline{x}$ embodies the observed exogenous variables with expected value $\underline{v}$ and error $\underline{w}$. The true exogenous variables are represented by $\underline{\xi}$ which is related to the observed variables by the coefficient matrix Γ.

Structural Model

$$\underline{\eta} = A\underline{\eta} + B\underline{\xi} + \underline{\theta} \qquad (25)$$

The definitions for $\underline{\eta}$ and $\underline{\xi}$ remain as before. The vector of θ's contains the equation errors while A and B are the structural coefficient matrices relating the true endogenous and exogenous variables to the true endogenous variables.

To fully understand the importance of each of these components we must examine the structure of the variance-covariance matrix of $\underline{y}$ and $\underline{x}$. In connection with this effort we must define the following additional parameter matrices:

Σ = the variance-covariance matrix of $\underline{y}$ and $\underline{x}$ composed of Σ_x, Σ_y, and Σ_{xy};

$\Sigma_{\underline{\xi}}$ = the variance-covariance matrix of the latent exogenous variables;

Σ_θ = the variance-covariance matrix of the errors in equations;

Ψ_ε = the variance-covariance matrix of the measurement errors associated with $\underline{y}$;

Ψ_ω = the variance-covariance matrix of the measurement errors associated with $\underline{x}$.

In addition to these definitions, several assumptions are made. The measurement errors, $\underline{\varepsilon}$ and $\underline{\omega}$, are assumed to be uncorrelated with each other and with the latent variables, $\underline{\eta}$ and $\underline{\xi}$. Finally, the residual errors, $\underline{\theta}$, are uncorrelated with the true exogenous variables, $\underline{\xi}$.

For convenience we may express the variance-covariance matrix of $\underline{y}$ and $\underline{x}$ in partitioned form:

$$\Sigma = \begin{bmatrix} \Sigma_y & \Sigma_{yx} \\ \Sigma_{xy} & \Sigma_x \end{bmatrix}$$

where

$\Sigma_{yx} = \Sigma'_{xy}$.

Given the preceding definitions and assumptions, each of the components of Σ may be expressed in terms of the parameters discussed as follows:

$$\Sigma_y = \Lambda[(I-A)^{-1}(B\Sigma_\zeta B' + \Sigma_\theta)(I-A)^{-t}]\Lambda' + \Psi_\varepsilon \quad (26)$$

$$\Sigma_x = \Gamma\Sigma_\zeta\Gamma' + \Psi_\omega \quad (27)$$

$$\Sigma_{xy} = \Gamma\Sigma_\zeta B'(I-A)^{-t}\Lambda'. \quad (28)$$

Estimation

If we assume that the composite vector = ($\underline{x}'$, $\underline{y}'$) is distributed multivariate normal with a variance-covariance matrix as expressed above, the various parameters of the overall model may be estimated via the maximum likelihood method. The values of the parameters which maximize the effective part of the log likelihood function,

$$\text{Log } L = - [(N-1)/2][\log|\Sigma| + tr(S\Sigma^{-1})] \quad (29)$$

are the maximum likelihood estimators. They may be found by taking the partial derivatives of the log likelihood function with respect to each of the parameters in the model, equating them to zero, and simultaneously solving the resulting equations. Since the explicit solution is obtainable only for a few restricted versions of the general model, some numerical solution must be employed in actual practice. The particular approach taken by Jöreskog and Van Thillo (1972) employs two numerical methods, the method of steepest descent and the Davidon-Fletcher-Powell method. The first approach is used to generate an approximate solution for the parameters in the neighborhood of the actual solution, while the second produces the final solution.

Chapter 4

General Structural Equation Model for Hierarchical Data

Following the schema established previously with the presentation of Jöreskog's Linear Structural Equation System Model, the structural equation model for hierarchical data is set out below. To facilitate the presentation, a notational convention has been adopted whereby a variable or parameter associated with the subject's within-groups level stands alone and the corresponding parameter at the between-groups level is subscripted with a lower case b. This should serve to preserve the conceptual similarities between Jöreskog's and this model while highlighting their differences.

As with Jöreskog's model, the new model may be set forth as two related components: the measurement model and the structural model.

Measurement Model

$$\underline{y} = \underline{\mu} + \Lambda\underline{\eta} + \Lambda_b\underline{\eta}_b + \underline{\varepsilon} + \underline{\varepsilon}_b \qquad (30)$$

$$\underline{x} = \underline{v} + \Gamma\underline{\zeta} + \Gamma_b\underline{\zeta}_b + \underline{\omega} + \underline{\omega}_b \qquad (31)$$

The $\underline{\mu}$ and $\underline{v}$ vectors are simply the expected values of $\underline{y}$ and $\underline{x}$ respectively and are conceptually the same as the corresponding terms in Jöreskog's model. The matrix Λ contains coefficients relating the latent endogenous within-groups variables $\underline{\eta}$ to the observed variables, $\underline{y}$. Likewise, Λ_b serves to relate the true endogenous between-groups variables, $\underline{\eta}_b$, to the observed $\underline{y}$. The vectors $\underline{\varepsilon}$ and $\underline{\varepsilon}_b$ represent the errors of measurement associated with the within- and between-groups levels respectively. The coefficient matrices, Γ and Γ_b, and the vectors $\underline{\zeta}$, $\underline{\zeta}_b$, $\underline{\omega}$ and $\underline{\omega}_b$ bear similar relationships to the observed $\underline{x}$ vector.

Structural Model		Reduced Form of Structural Model	
$\underline{\eta} = A\underline{\eta} + B\underline{\zeta} + \underline{\theta}$	(32)	$(I-A)\underline{\eta} = B\underline{\zeta} + \theta$	(32′)
$\underline{\eta}_b = A_b\underline{\eta}_b + B_b\underline{\zeta}_b + \underline{\theta}_b$	(33)	$(I-A_b)\eta_b = B_b\underline{\zeta}_b + \theta_b$	(33′)

The first equation stipulates that the latent within-groups endogenous variables are expressible as linear functions of themselves (as determined by the coefficients in the A matrix) and the latent within-groups exogenous variables (as determined by the coefficients in the B matrix). Finally, we have the vector $\underline{\theta}$ containing the errors in equations associated with this part of the structural model. The second equation is composed of parallel constructs dealing with the expression of the between-groups latent endogenous variables.

As with Jöreskog's model there are a number of variance-covariance terms associated with these vector-valued variables. They are as follows:

$\Sigma_{\underline{\zeta}}$ - the variance-covariance matrix of the latent within-groups exogenous variables, $\underline{\zeta}$;

$\Sigma_{\underline{\theta}}$ - the variance-covariance matrix of the within-groups errors in equations, $\underline{\theta}$;

$\Psi_{\underline{\varepsilon}}$ - the variance-covariance matrix of the within-groups measurement error associated with the observed endogenous variables, $\underline{y}$;

$\Psi_{\underline{\omega}}$ - the variance-covariance matrix of the within-groups measurement error associated with the observed exogenous variables, $\underline{x}$;

$\Sigma_{\underline{\zeta}_b}$ - the variance-covariance matrix of the latent between-groups exogenous variables, $\underline{\zeta}_b$;

$\Sigma_{\underline{\theta}_b}$ - the variance-covariance matrix of the between-groups errors in equations, $\underline{\theta}_b$;

$\Psi_{\underline{\varepsilon}_b}$ - the variance-covariance matrix of the between-groups measurement error associated with the observed endogenous variables;

$\Psi_{\underline{\omega}_b}$ - the variance-covariance matrix of the between-groups measurement error associated with the observed exogenous variables.

If, as before, we assume the measurement errors to be uncorrelated with each other and with the latent variables and that the residual errors are uncorrelated with the true exogenous variables and that all variables at one level are uncorrelated with those at another, the variance-covariance matrix of the <u>observed variables</u> can be expressed as a function of the parameters defined above.

Let the combined vector of observed scores for an individual be represented by the vector $\underline{z}$ where

$$\underline{z} = \begin{bmatrix} \underline{x} \\ - - - \\ \underline{y} \end{bmatrix}$$

The variance-covariance matrix among these observed variables, $V(\underline{z})$, can then be represented as Σ_z and we have

$$\Sigma_z = \Sigma + \Sigma_b \tag{34}$$

The parametric composition of Σ and Σ_b is then:

$$\Sigma = \left[\begin{array}{c|c} \Gamma\Sigma_\zeta\Gamma' + \Psi_\omega & \Gamma\Sigma_\zeta B'(I-A)^{-t}\Lambda' \\ \hline \Lambda(I-A)^{-1}B\Sigma_\zeta\Gamma' & \begin{array}{c} \Lambda[(I-A)^{-1}B\Sigma_\zeta B'(I-A)^{-t}]\Lambda' \\ + \\ \Lambda[(I-A)^{-1}\Sigma_\theta(I-A)^{-t}]\Lambda' + \Psi_\varepsilon \end{array} \end{array}\right] \tag{35}$$

$$\Sigma_b = \left[\begin{array}{c|c} \Gamma_b\Sigma_{\zeta_b}\Gamma_b' + \Psi_{\omega_b} & \Gamma_b\Sigma_{\zeta_b}B_b'(I-A_b)^{-t}\Lambda_b' \\ \hline \Lambda_b(I-A_b)^{-1}B_b\Sigma_{\zeta_b}\Gamma_b' & \begin{array}{c} \Lambda_b[(I-A_b)^{-1}B_b\Sigma_\zeta B_b' (I-A_b)^{-t}]\Lambda_b' \\ + \\ \Lambda_b[(I-A_b)^{-1}\Sigma_{\theta_b} (I-A_b)^{-t}]\Lambda_b' + \Psi_{\varepsilon_b} \end{array} \end{array}\right] \tag{36}$$

Parameter Estimation: General Considerations

If we assume the overall vector, $[\underline{y}', \underline{x}']$, to be multivariate normally distributed, the parameters in the measurement and structural components may be estimated by use of the maximum likelihood principle. We have, from Schmidt, the effective part of the log likelihood function of the hierarchical situation under consideration:

$$F = \frac{m-mn}{2}\log(|\Sigma|) - \frac{m}{2}\log(|\Sigma + n\Sigma_b|) - \frac{mn}{2}\,\mathrm{tr}\,\{\Sigma^{-1}S\} - \frac{n}{2}\,\mathrm{tr}\,\{(\Sigma+n\Sigma_b)^{-1}S_b\}. \tag{37}$$

After replacing Σ and Σ_b with the expressions set forth above we need but to choose values for the parameters which maximize F. This may be accomplished by, first, taking the partial derivatives of F with respect to each of the elements in the parameter matrices. The resulting first derivatives are set equal to zero and simultaneous solutions found for the parameters. Unfortunately, these equations are even more complicated than those discussed previously; therefore, some numerical solution is called for.

The first derivatives are fully set forth in Appendix A. In general, they were found by, first, taking the partial derivatives of F with respect to Σ and Σ_b, and the partial derivaties of Σ and Σ_b with respect to their respective parameter matrices. The application of the chain rule for matrix derivatives completed the process.

To carry out the numerical solution for the maximum likelihood estimates, the same procedure used by Jöreskog was employed. The method of steepest descent serves as a first stage in the estimation procedure until the estimated values for the solution approach a reasonable neighborhood to the actual solution. The Fletcher-Powell method is then used to accomplish the final maximization of the likelihood function.

Numerical Solutions for Parameter Estimates

The most satisfying approach to generating parameter estimates would be to find simple analytical expressions for the parameters using the "normal" equations arrived at by setting the first derivatives of the log likelihood function equal to zero. The complexity of these expressions, however, is such that straightforward solutions are possible only in the case of very simplified models. Instead, we must turn to the use of numerical techniques whereby parameter estimates are individually generated for each model and each set of data.

A quick perusal of any text that touches on non-linear programming (for instance--Luenberger [1965]) reveals a wealth of techniques whereby solutions may be obtained for systems of equations such as those in the present instance. The key criteria in the selection of one or more of these techniques for a particular application seem to be global convergence and rate of convergence. The first of these criteria refers to the ability of an algorithm to arrive at a "true" solution irrespective of the point at which the algorithm starts. The second has to do with the number of iterations required to arrive at a solution. While the first constitutes a necessary condition for the choice of a particular method, the second determines the efficiency of the estimation routine.

The general approach adopted by most numerical solution algorithms involves a series of steps outlined below:

1) Choose an initial value for the solution, X_0.
2) Determine the direction in which the solution is to be modified.
3) Choose as the new value for the solution, X_1, the point at which the function is minimized in the direction determined by step 2.
4) Compare $f(X_0)$ with $f(X_1)$ to see if a significant change has been made.
5) If nothing has appreciably changed, the solution has been found.
6) If changes have been made, return to step 2 and continue.

The primary difference which characterizes the various methods is the way in which the direction of modification is determined.

In the present instance the technique actually implemented involves a combination of two fairly widely used approaches, the method of steepest descent and the Davidon-Fletcher-Powell method. According to Jöreskog (1969) the first of these approaches offers rapid advances toward the immediate neighborhood of the solution followed by relatively slower convergence upon the solution. The second method, on the other hand, is relatively slow in arriving at the neighborhood of the solution but fast thereafter. The algorithm employed relies upon a number of steepest descent iterations ceasing when the change in the value of the function is less than five percent from one iteration to the next. These are then followed by the application of the Davidon-Fletcher-Powell method to arrive at a fully-converged solution. The operation of each is set forth below along with that of Newton's method on which Davidon-Fletcher-Powell is based.

Steepest Descent

If we designate f as the function which we wish to minimize having continuous first partial derivatives ∂f, then the method of steepest descent directs us to take as the k+1 value for our parameter estimates the following:

$$X_{k+1} = X_k - \alpha_k \partial f(X_k) \tag{38}$$

where α_k is a positive number which minimizes $f(X_k - \alpha_k \partial f(X_k))$. Repeated application of this method will yield values for X which correspond to the solution sought.

Newton's Method

In its pure form, Newton's method involves a change in the iterative approach involved in the steepest descent technique whereby the direction for solution modification is found. The net result is the use of the following expression for X_{k+1}:

$$X_{k+1} = X_k - F(X_k)^{-1} \partial f(X_k)' \tag{39}$$

where $F(X_k)$ is the matrix of second derivatives of f evaluated at the point X_k.

Since global convergence cannot be assured for this method, its typical operationalization is usually of the form

$$X_{k+1} = X_k - \alpha_k F(X_k)^{-1} \partial f(X_k)' \tag{40}$$

which has global convergence properties. In addition, use of this method yields convergence requiring fewer cycles than does the method of steepest descent assuming X_0 is chosen sufficiently close to the actual solution. The only drawback to applying this method is the need to constantly reevaluate $F(X_k)^{-1}$, a process which can be quite time consuming.

Davidon-Fletcher-Powell

This approach belongs to a class of quasi-Newton methods all of which are characterized by the use of approximations to the inverse of the matrix of second partial derivatives. This particular method involves starting the minimization procedure with both an initial estimate for the solution, X_0, and an initial estimate of the inverse of the matrix of second derivatives, S_0. Successive approximations to the solutions are found by employing the following equation:

$$X_{k+1} = X_k - \alpha_k S_k \, \partial f(X_k) \tag{41}$$

where, as with the method of steepest descent and Newton's modified method, α_k is a positive number which minimizes $f(X_k - \alpha_k S_k \, \partial F(X_k))$. Successive approximations to the inverse of the matrix of second derivatives are found through the use of the following relationship:

$$S_{k+1} = S_k + \frac{p_k p_k'}{p_k' q_k} - \frac{S_k q_k q_k' S_k}{q_k' S_k q_k} \tag{42}$$

where p_k designates the difference between X_k and X_{k+1} and q_k is equal to the difference between $\partial f(X_k)$ and $\partial f(X_{k+1})$.

Both of the latter two methods may fail to converge to the appropriate solution given initial values of X_0 which depart too much from the actual solution. This absence of guaranteed global convergence motivates the chaining of the method of steepest descent with that of Davidon-Fletcher-Powell.

Identifiability

For a particular model to be of any real use, we must be able to estimate the parameters associated with that model. For the parameters to be estimable from a particular set of data two conditions must be met. The first is that Σ and Σ_b must be of full rank. This will be the case if enough observations are taken within each unit and if enough units are observed. One must also avoid the inclusion of variables which are linearly dependent upon other variables. In practice, the invertability of the unrestricted maximum likelihood estimates for Σ and Σ_b guarantees that this condition is met.

The other, and far more difficult to determine, condition for the estimability of a set of parameters is that the model be identified. The identifiability of a model basically means that, for two distinct models to give rise to the same Σ and Σ_b, their parameters must be identical in all respects. In other words, the parameters of an identified model must be unique. It must be emphasized that this condition is on the model in question and has nothing to do with a particular set of data.

When dealing with regression models, the only way in which a model may be under-identified is if one or more of the predictor variables is linearly dependent upon the others. This situation is readily noted due to the fact that the $X'X$ matrix has no unique inverse even though more observations were taken than the number of predictor variables. While being a condition easily detected, the remedy may not be so easy without considerable thought on the definitions of the predictor variables.

With more complex models such as the ones addressed in this paper, determining if a specific model is identifiable may be much more difficult. Econometricians have addressed this problem extensively and, for a variety of models more complex than the simple regression model, have formulated some mathematical rules for identifying necessary and sufficient conditions for model identifiability (see Fisher [1970 and 1966], for instance).

The work that comes closest to addressing model identifiability in a situation similar to that currently considered is represented by Geraci

(1977). In this paper, he provides an algorithm by which the identifiability of a particular uni-level model might be established. Although he restricted the model considered to have no measurement error or complex factor structure, establishing the criterion for model identifiability involved the solution of a set of equations hardly less formidable than those involved in the system whose identifiability was of interest.

Jöreskog (1977) suggests that as a necessary condition for the identifiability of a particular model that the number of unknown elements be less than $\frac{1}{2}(p + q)(p + q + 1)$. While this must be true for a unique solution to exist, it by no means guarantees that one does. Given the complexity of the current model in the face of the rather complicated necessary and sufficient condition advanced by Geraci when dealing with a much simpler model, it is no real surprise that a straightforward test for the identity of a particular model of the sort considered here is difficult to achieve.

Wiley (1973) in considering the identification problem in a uni-level model of the same form as the one considered here offers a very useful suggestion. If a program were available to compute a numerical estimate of the information matrix for the parameters and if some reasonable estimates for the parameters were inserted into the model, then model identifiability could be reasonably assumed if the information matrix was of full rank. The benefit from adopting this approach is that the identifiability of a particular model could be reasonably assured prior to the estimation of its free parameters.

Estimating Parameters in the General Model

While there exist a variety of methods whereby estimates of the parameters of the general model might be derived including both unweighted and generalized least squares, the most straightforward in terms of estimation, tests of fit and producing asymptotic standard errors of the estimates is the method of maximum likelihood. Parameter estimates produced by this method are those values for the parameters which maximize the likelihood of the observed data given an assumed underlying distribution where the likelihood of a particular set of observations is their joint probability given the parameter values.

Given an expression for the joint probability of a sample of observed values, values for the parameters may be formed by first taking the derivatives of the log likelihood function with respect to the parameters themselves, equating these to zero, and finally, solving the resulting set of simultaneous equations. The key to the entire process is the formulation of the likelihood function.

Nearly all of the literature reviewed which dealt with the maximum likelihood estimation of structural equations or analysis of covariance structures addressed itself to the situation where a single sample of observations was drawn from a presumed multivariate normal distribution. Under those circumstances, the effective part of the logarithm of the likelihood function has the general form

$$M = \mathrm{tr}\,(\Sigma^{-1}S) - \log|\Sigma|. \tag{43}$$

Only the work carried out by Schmidt considered the situation involving a two-stage sampling process where observations were sampled from primary sampling units which themselves were sampled. Under the assumption of doubly multivariate normally distributed observations where variables for each individual observation are multivariate normally distributed and observations themselves are similarly distributed within groups, Schmidt derived the following expression for the logarithm of the effective part of the likelihood function:

$$\log L = \frac{m-mn}{2} \log |\Sigma| - \frac{m}{2} \log |\Sigma + n\,\Sigma_b| - \frac{mn}{2} \operatorname{tr} \{\Sigma^{-1} S\} - \frac{m}{2} \operatorname{tr} \{(\Sigma + n\,\Sigma_b)^{-1} S_b\} \qquad (44)$$

where m is the number of groups, n the number of secondary units within each of the m primary units, S and S_b are the within- and between-groups observed variance-covariance matrices respectively, and Σ and Σ_b are the underlying variance-covariance matrices for the within- and between- groups levels respectively. This expression served as the basis for the estimation procedure implemented here.

The next step in producing the maximum likelihood estimates for the parameters in the general model calls for expressions for the first derivatives of the log likelihood function with respect to each of the parameter matrices in the general model. The simplest way to arrive at such expressions is through the use of the chain rule for derivatives involving matrices. According to McDonald and Swaminathan (undated), if the elements of a matrix Z are functions of the elements of a matrix Y which are themselves functions of another matrix X, the partial derivative of Z with respect to X can be expressed as:

$$\frac{\partial Z}{\partial X} = \frac{\partial Y}{\partial X} \frac{\partial Z}{\partial Y} .$$

This is also true if Z is some scalar function of X.

Since the log likelihood function is a function of but two matrices, Σ and Σ_b, each of which is a function of a subset of the general parameters of the model, the partial derivatives of the log likelihood function with respect to any one parameter matrix, say C, can be conveniently expressed as follows:

$$\frac{\partial \log \ell}{\partial C} = \begin{cases} \frac{\partial \Sigma}{\partial C} \frac{\partial \log \ell}{\partial \Sigma}, & \text{if C is a parameter at the within-groups level} \\ \text{or} & \\ \frac{\partial \Sigma_b}{\partial C} \frac{\partial \log \ell}{\partial \Sigma_b}, & \text{if C is a parameter at the between-groups level.} \end{cases}$$

Schmidt has derived expressions for the rightmost partial derivatives of each equation. These expressions are as follows:

$$\begin{aligned} \frac{\partial \log \ell}{\partial \Sigma} = \{ & m(1-n)\, \Sigma^{-1} - m(\Sigma + n\, \Sigma_b)^{-1} + mn\, \Sigma^{-1}\, S\Sigma^{-1} \\ & + (\Sigma + n\, \Sigma_b)^{-1}\, S_b\, (\Sigma + n\, \Sigma_b)^{-1}\} - \tfrac{1}{2}\, \text{diag}\, \{m\, (1-n)\, \Sigma^{-1} \\ & - m\, (\Sigma + n\, \Sigma_b)^{-1} + mn\, \Sigma^{-1}\, S\Sigma^{-1} \\ & + (\Sigma + n\, \Sigma_b)^{-1}\, S_b\, (\Sigma + n\, \Sigma_b)^{-1}\} \end{aligned} \tag{45}$$

$$\begin{aligned} \frac{\partial \log \ell}{\partial \Sigma_b} = \{ & mn\, (\Sigma + n\, \Sigma_b)^{-1}\, S_b\, (\Sigma + n\, \Sigma_b)^{-1} - mn\, (\Sigma + n\, \Sigma_b)^{-1}\} \\ & - \tfrac{1}{2}\, \text{diag}\, \{mn\, (\Sigma + n\, \Sigma_b)^{-1}\, S_b\, (\Sigma + n\, \Sigma_b)^{-1} \\ & - mn\, (\Sigma + n\, \Sigma_b)^{-1}\} \end{aligned} \tag{46}$$

where S corresponds to the pooled within-groups observed variance-covariance matrix and S_b corresponds to n times the between-groups variance-covariance matrix.

The remaining components necessary to complete the expressions for the partial derivatives of the log likelihood function with respect to the parameter matrices in the general model are the partial derivatives of Σ and Σ_b with respect to the parameter matrices involved in each. These expressions have been derived through the use of matrix calculus and are set forth fully in Appendix A.

Standard Error Estimation and Test of Fit of the Estimated Model

Just as the maximum likelihood principle leads directly to the estimation of model parameters through the use of first order partial derivatives, the second order partial derivatives assist in the estimation of standard errors for the parameters involved. It has been shown that the negative inverse of the expected values of the matrix of second partial derivatives is equal to the asymptotic covariance matrix of the maximum likelihood estimators. This may be simply expressed as follows:

$$V(\hat{\underline{\theta}}) = - E \left[\frac{\partial^2 \log \ell}{\partial \theta_i \partial \theta_j} \right]^{-1} . \qquad (47)$$

The square roots of the diagonal values of this matrix yield estimates for the standard errors of their associated parameters. Since maximum likelihood estimators are, for sufficiently large numbers of observations, normally distributed, the estimated standard errors may be used to establish confidence intervals about the parameters estimated and thus provide statistical tests for the parameter values against any particular null hypothesis of interest. While the tests would not be strictly independent of one another for a given model and set of data, they will yield useful information toward the refinement of a particular model.

Schmidt (1969) has shown that, for the hierarchical situation of interest in the current investigation, the expected value of the matrix of second partial derivatives of the log likelihood function is a function of both the first and second derivatives. Furthermore, it can be expressed by the following formula for the general ijth element:

$$E\left[\frac{\partial^2 \log \ell}{\partial\theta_i \partial\theta_j}\right] = -\frac{1}{2}\,\mathrm{tr}\left\{\frac{\partial^2(\Sigma + n\Sigma_b)}{\partial\theta_i\partial\theta_j}\,(\Sigma + n\Sigma_b)^{-1}\right\} \tag{48}$$

$$+\frac{m(1-n)}{2}\,\mathrm{tr}\left\{(\Sigma + n\,\Sigma_b)^{-1}\,\frac{\partial(\Sigma + n\,\Sigma_b)}{\partial\theta_i}\,(\Sigma + n\,\Sigma_b)^{-1}\,\frac{\partial(\Sigma + n\,\Sigma_b)}{\partial\theta_j}\right\}.$$

Expressions for the first and second derivatives of Σ and Σ_b with respect to individual elements of the parameter matrices in the general model have been derived and are set forth in Appendices B and C. So as to conserve space, only the nonredundant expressions are shown. Since the order in which the partial derivatives are taken has no effect upon their value, only the unique formulae are shown.

When all of the various elements involved in a given parameterization have been calculated and assembled in matrix form, the negative inverse of this matrix estimates the covariance matrix of the estimators. The documentation for a computer program implementing this procedure is included in Appendix D and its listing is included in Appendix E.

While the foregoing provides a means whereby confidence intervals may be established about individual parameter estimates in a particular model, it does not actually enable the testing of a model as a complete entity. To this end, we must turn to yet another construct derived from the maximum likelihood principle, the likelihood ratio.

To generate parameter estimates for a particular model and set of data of interest, we choose as our estimates those values of the parameters in the model which yield the largest value of the likelihood function given the data at hand. Under some other parameterization of the model both the estimates and the value of the likelihood functions would likely differ when employing the same set of data. In particular, we can posit as our alternative model one which is least restrictive in that it will yield the largest value for the likelihood ratio. This model simply asserts that the data arise from a multivariate normal distribution with parameters Σ and Σ_b with no further parameterization placed on these two matrices. Thus our estimates of Σ and Σ_b are unrestricted by any constraints placed upon them and the value of the likelihood function so obtained can be referred to as the maximum value of the likelihood function over the unrestricted parameter space.

Under any other particular parameterization of Σ and Σ_b furnished by our model, the maximum of the likelihood function can be referred to as the maximum over the restricted parameter space and cannot be larger than the maximum over the unrestricted space. This implies that the ratio of the latter to the former has as its maximum value 1 and, since neither term can take on anything other than non-negative values, as its minimum 0. This quantity is known as the likelihood ratio and provides a means whereby the fit of a particular model (i.e., the ability of a model to replicate Σ and Σ_b) may be evaluated. Since the likelihood ratio is based upon two random variables (the maximum of the likelihood function over the restricted and unrestricted parameter space) it too is a random variable. In addition, for large sample size, the negative value of twice the logarithm of the likelihood ratio has approximately the chi-square distribution. Thus, we have as our test statistic the following:

$$\chi^2 = -2 \log (L_{restricted}/L_{unrestricted}). \tag{49}$$

This is readily seen to be equivalent to the more convenient expression:

$$\chi^2 = 2(\log L_{unrestricted} - \log L_{restricted}) \tag{50}$$

The degrees of freedom associated with it are equal to the difference between the number of unique elements in Σ and Σ_b and the number of unique parameters estimated in the restricted model.

Larger values of the test statistic which lie far to the right on the reference distribution are unlikely under the assumption that the model fits the data. Thus, likelihood ratio statistics of low probability under the assumption of model fit point to overall weaknesses in the model, the particulars of which should be addressed through inspection of the asymptotic standard errors and the discrepancies between the unrestricted and restricted estimates for Σ and Σ_b.

Chapter 5

Applications

Analysis of Artificial Data: Testing the Estimation Procedure Using a Simple Model

As a part of the work carried out by Schmidt (1969), several sets of data with a predetermined structure were generated. These data sets were then analyzed using four different parameterizations, one of which reflected the true structure of the data. As one test of the estimation routine currently implemented, one of these data sets was reanalyzed making use of the same parameterizations employed by Schmidt. The S and S_b matrices used as input to the estimation routine are displayed in Figure 1.

Due to the fact that the model considered by Schmidt did not explicitly allow for the presence of exogenous variables, only the portion of the current model dealing with the interrelationships among endogenous variables could be examined. This restricted model parameterizes the within- and between-groups variance-covariance matrices as follows:

$$S = \Lambda \Sigma_\theta \Lambda' + \Psi_\varepsilon \quad (51)$$

$$S_b = \Lambda_b \Sigma_{\theta_b} \Lambda_b' + \Psi_{\varepsilon_b} \quad . \quad (52)$$

Thus, the matrices associated with exogenous variables (Γ, Γ_b, B_b, Σ_{ζ_b}, B, Σ_ζ, Ψ_ω, and Ψ_{ω_b}) were omitted from the model. Additionally, the elements of A and A_b were fixed to zero, while Λ and Λ_b were both

$$S = \begin{bmatrix} 12.114 & & & \\ 2.607 & 12.236 & & \\ 4.620 & 2.390 & 13.685 & \\ 1.701 & 5.265 & 5.382 & 16.154 \end{bmatrix}$$

$$S_b = \begin{bmatrix} 161.350 & & & \\ 121.387 & 312.413 & & \\ 40.613 & 38.965 & 161.626 & \\ 63.700 & 64.280 & 41.990 & 157.749 \end{bmatrix}$$

$n = 10$

$m = 50$

Figure 1. **Artificial Data Obtained from Schmidt (1969).**

equated to the following design matrix:

$$\Lambda = \Lambda_b = \begin{bmatrix} 1.0 & .5 & .5 \\ 1.0 & .5 & -.5 \\ 1.0 & -.5 & .5 \\ 1.0 & -.5 & -.5 \end{bmatrix}$$

The resulting model was, therefore, a function of but four parameter matrices, Σ_θ, Σ_{θ_b}, Ψ_ε, and Ψ_{ε_b}. The various forms of these matrices for which parameters were estimated are presented in Table 1. The true model, that which actually gave rise to the data in question, is Model 1 for which Σ_θ and Σ_{θ_b} are diagonal matrices while Ψ_ε and Ψ_{ε_b} are heterogeneous.

The results obtained from the estimation routine using the first parameterization of the model where the Σ_θ matrices were constrained to be diagonal and the diagonal elements of the Ψ_ε matrices were allowed to be heterogeneous are set forth in Table 2. Corresponding estimates obtained from Schmidt's work are presented alongside those from the new estimation routine. The associated asymptotic standard errors obtained from the implementation of the procedure for estimating standard errors are also presented in the same table, as is the chi-square value and degrees of freedom associated with the model.

A comparison of the estimates obtained from the current program and that developed by Schmidt reveals that the results are identical to at least two decimal places. Differences beyond this point are attributable to the accuracy of the calculations required to obtain S and S_b from Schmidt's work. The obtained chi-square values for the test of fit

Table 1

Parameterizations Employed in Analyzing Artificial Data from Schmidt

Model	Σ_θ and Σ_{θ_b}	ψ_ε and ψ_{ε_b}
1	diagonal	heterogeneous
2	diagonal	heterogeneous
3	general	heterogeneous
4	general	heterogeneous

Table 2

Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 1

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{11}}$	4.875	4.875	.211
$\Sigma_{\zeta_{21}}$	--	--	--
$\Sigma_{\zeta_{22}}$	4.075	4.075	.821
$\Sigma_{\zeta_{31}}$	--	--	--
$\Sigma_{\zeta_{32}}$	--	--	--
$\Sigma_{\zeta_{33}}$	6.401	6.403	.984
$\Psi_{\varepsilon_{11}}$	6.959	6.957	.613
$\Psi_{\varepsilon_{22}}$	6.569	6.570	.632
$\Psi_{\varepsilon_{33}}$	7.129	7.127	.648
$\Psi_{\varepsilon_{44}}$	9.376	9.377	.699
$\Sigma_{\zeta_{b_{11}}}$	7.013	7.014	3.353
$\Sigma_{\zeta_{b_{21}}}$	--	--	--
$\Sigma_{\zeta_{b_{22}}}$	6.840	6.842	10.923

(continued)

Table 2 (continued)

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{b_{31}}}$	--	--	--
$\Sigma_{\zeta_{b_{32}}}$	--	--	--
$\Sigma_{\zeta_{b_{33}}}$	.000	.000	5.258
$\Psi_{\varepsilon_{b_{11}}}$	3.694	3.693	4.588
$\Psi_{\varepsilon_{b_{22}}}$	6.713	6.713	5.327
$\Psi_{\varepsilon_{b_{33}}}$	11.082	11.086	8.679
$\Psi_{\varepsilon_{b_{44}}}$	7.662	7.661	8.202
χ^2/df	17.4778 6	17.5 6	

of the model are identical within rounding error. Additionally, the non-zero parameter estimates all differ from zero by more than one standard error, as would be hoped for, given that the form of the estimated model corresponds to that employed in generating Schmidt's data.

Parallel results with respect to parameter estimates, asymptotic standard errors, and chi-square statistics for the tests of fit of the remaining three models are contained in Tables 3 through 5. The parameter estimates obtained from the implementation of the present, more general model are nearly identical to those reported by Schmidt as are the chi-square statistics for each model. Since the asymptotic standard errors reported by Schmidt were obtained as a by-product of the Fletcher-Powell algorithm and not from the evaluation of the expected value of the matrix of second derivatives, they are not reported here; however, where comparable values were computed, the standard errors were of similar magnitude.

The results of these analyses offer evidence that the currently implemented estimation procedures perform accurately with models of at least the complexity of those considered earlier by Schmidt. A more comprehensive test of the accuracy of the estimation procedure required data arising from a model with a more complex structure. The next section presents results from the analysis of data with such a complex structure.

Table 3

Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 2

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{11}}$	4.965	4.965	.216
$\Sigma_{\zeta_{21}}$	--	--	--
$\Sigma_{\zeta_{22}}$	4.329	4.330	.835
$\Sigma_{\zeta_{31}}$	--	--	--
$\Sigma_{\zeta_{32}}$	--	--	--
$\Sigma_{\zeta_{33}}$	6.430	6.433	.990
$\Psi_{\varepsilon_{11}}$	7.398	7.396	.139
$\Psi_{\varepsilon_{22}}$	7.398	7.396	.139
$\Psi_{\varepsilon_{33}}$	7.398	7.396	.139
$\Psi_{\varepsilon_{44}}$	7.398	7.396	.139
$\Sigma_{\zeta_{b_{11}}}$	6.259	6.259	3.309
$\Sigma_{\zeta_{b_{21}}}$	--	--	--
$\Sigma_{\zeta_{b_{22}}}$	5.833	5.834	11.431

(continued)

Table 3 (continued)

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{b_{31}}}$	--	--	--
$\Sigma_{\zeta_{b_{32}}}$	--	--	--
$\Sigma_{\zeta_{b_{33}}}$	.000	.000	6.315
$\Psi_{\varepsilon_{b_{11}}}$	7.717	7.717	1.880
$\Psi_{\varepsilon_{b_{22}}}$	7.717	7.717	1.880
$\Psi_{\varepsilon_{b_{33}}}$	7.717	7.717	1.880
$\Psi_{\varepsilon_{b_{44}}}$	7.717	7.717	1.880
χ^2/df	28.9121 / 12	28.9 / 12	

Table 4

Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 3

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{11}}$	4.964	4.964	.222
$\Sigma_{\zeta_{21}}$	-1.541	-1.542	.216
$\Sigma_{\zeta_{22}}$	4.324	4.326	.850
$\Sigma_{\zeta_{31}}$	-.356	-.357	.235
$\Sigma_{\zeta_{32}}$	.755	.755	.412
$\Sigma_{\zeta_{33}}$	6.417	6.418	1.053
$\Psi_{\varepsilon_{11}}$	7.328	7.329	.908
$\Psi_{\varepsilon_{22}}$	7.508	7.504	.796
$\Psi_{\varepsilon_{33}}$	6.744	6.747	.810
$\Psi_{\varepsilon_{44}}$	8.020	8.019	1.323
$\Sigma_{\zeta_{b_{11}}}$	6.342	6.342	3.364
$\Sigma_{\zeta_{b_{21}}}$	4.084	4.083	2.644
$\Sigma_{\zeta_{b_{22}}}$	6.142	6.145	11.792

(continued)

Table 4 (continued)

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{b_{31}}}$	-.997	-.997	1.913
$\Sigma_{\zeta_{b_{32}}}$	-1.745	-1.747	5.410
$\Sigma_{\zeta_{b_{33}}}$	.503	.504	5.579
$\Psi_{\varepsilon_{b_{11}}}$	4.402	4.399	8.317
$\Psi_{\varepsilon_{b_{22}}}$	4.567	4.565	8.550
$\Psi_{\varepsilon_{b_{33}}}$	11.089	11.084	10.278
$\Psi_{\varepsilon_{b_{44}}}$	9.741	9.738	12.960
χ^2/df	.2635	.26	
	0	0	

Table 5

Parameter Estimates, Standard Errors, and Test of Fit for the Analysis of Schmidt's Data Using Model 4

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{11}}$	4.964	4.964	.222
$\Sigma_{\zeta_{21}}$	-1.533	-1.533	.169
$\Sigma_{\zeta_{22}}$	4.325	4.325	.850
$\Sigma_{\zeta_{31}}$	-.538	-.538	.187
$\Sigma_{\zeta_{32}}$	1.029	1.029	.235
$\Sigma_{\zeta_{33}}$	6.418	6.418	1.032
$\Psi_{\varepsilon_{11}}$	7.398	7.400	.198
$\Psi_{\varepsilon_{22}}$	7.398	7.400	.198
$\Psi_{\varepsilon_{33}}$	7.398	7.400	.198
$\Psi_{\varepsilon_{44}}$	7.398	7.400	.198
$\Sigma_{\zeta_{b_{11}}}$	6.349	6.349	3.244
$\Sigma_{\zeta_{b_{21}}}$	2.618	2.618	2.016
$\Sigma_{\zeta_{b_{22}}}$	6.258	6.258	11.391

(continued)

Table 5 (continued)

Parameter	Estimate From Current Program	Estimate From Schmidt's Work	Asymptotic Error Variance
$\Sigma_{\zeta_{b_{31}}}$	-.935	-.935	1.618
$\Sigma_{\zeta_{b_{32}}}$	-2.119	-2.119	2.670
$\Sigma_{\zeta_{b_{33}}}$	.718	.718	6.290
$\Psi_{\varepsilon_{b_{11}}}$	7.360	7.358	2.286
$\Psi_{\varepsilon_{b_{22}}}$	7.360	7.358	2.286
$\Psi_{\varepsilon_{b_{33}}}$	7.360	7.358	2.286
$\Psi_{\varepsilon_{b_{44}}}$	7.360	7.358	2.286
χ^2/df	7.3587 6	7.36 6	

Analysis of Artificial Data: Testing the Estimation Procedure Using a Complex Model

The first step in testing the program through the analysis of artificial data was to generate an S and S_b arising from a more complex structure. The underlying principle employed was to assign arbitrary values to the parameters in the general model and, from these values, produce within- and between-groups variance-covariance matrices. At least two methods are available for carrying this out. One method involves a two-stage process characterized by, first, generating observations from a multivariate normal distribution with the appropriate characteristics and, second, calculating the within- and between-groups variance-covariance matrices based on the artificial random observations from the first stage. This method is ideally suited to studies of the empirical distribution of the parameter estimates over repeated analyses of data with the same underlying structure using different samples of observation.

Since this was not a goal of the present investigation, it was determined that such an approach would be excessively laborious and time consuming. Instead, an alternative was adopted that more readily yielded analyzable data with a known structure, but did not rely on any stochastic processes. Arbitrary values were assigned to the parameters in the general model and the resulting matrices were mathematically combined according to the model to yield artificial underlying matrices, Σ and Σ_b. These were then placed in the equations for the unrestricted maximum likelihood solutions and values for S and S_b were obtained. Analysis of such data using the correctly specified model should result in parameter estimates that exactly match the original arbitrary values.

So as to simplify the calculation of the artificial data, the structure and parameters at both levels were defined to be identical. Since the model to be estimated did not explicitly constrain the corresponding within- and between-groups parameters to be equal, no unfair advantage was accorded to the program through the use of this convention. In the absence of such specified constraints, the program must still independently estimate the parameters at both the within- and between-groups levels.

The values assigned to each of the sixteen parameter matrices associated with the model are found in Figure 2. To additionally simplify calculating S and S_b, the number of observations within each group and the number of groups (m and n, respectively) were set to 100. The resulting values for Σ and Σ_b are set forth in Figure 3. These two matrices gave rise to the generated observed matrices S and S_b using the following formulae from Schmidt (1969):

$$E(S) = \frac{n-1}{n} \Sigma \tag{53}$$

$$E(S_b) = n \, \Sigma_b + \Sigma . \tag{54}$$

For the purpose considered herein the expectation operations can be disregarded and the relationships treated as simple equalities.

The resulting values for S and S_b are presented in Figure 4. When these data were analyzed using the programs operationalizing the previously described estimation procedure, 4 steepest descent iterations took place before the stopping criterion was reached after which 26 Fletcher-Powell iterations followed. The parameter estimates are displayed in Table 6, and duplicate the original generating values to at least three decimal places. The matrices S and S_b were duplicated with

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & .5 \end{bmatrix}$$

Λ & Λ_b

$$\begin{bmatrix} 1 & 0 \\ -.5 & 1 \end{bmatrix}$$

(I-A) & $(I-A_b)$

$$\begin{bmatrix} .2 & .4 \\ .3 & .8 \end{bmatrix}$$

B & B_b

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

Σ_ζ & Σ_{ζ_b}

$$\begin{bmatrix} .2 & 0 \\ 0 & .03 \end{bmatrix}$$

Σ_θ & Σ_{θ_b}

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & .1 \end{bmatrix}$$

Ψ_ε & Ψ_{ε_b}

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & .5 \end{bmatrix}$$

Γ & Γ_b

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & .1 \end{bmatrix}$$

Ψ_ω & Ψ_{ω_b}

Figure 2. Parameter Values Used to Generate Variance-Covariance Matrices for Example II.

$$\Sigma = \Sigma_b = \begin{bmatrix} .40 & & & & & \\ .58 & 1.24 & & & & \\ .29 & .62 & .41 & & & \\ .20 & .40 & .20 & 1.00 & & \\ .40 & 1.00 & .50 & 0.00 & 1.00 & \\ .20 & .50 & .25 & 0.00 & .50 & .35 \end{bmatrix}$$

Figure 3. Values of Σ and Σ_b for Artificial Data Example II.

$$S = \begin{bmatrix} .3960 & & & & & \\ .5742 & 1.2276 & & & & \\ .2871 & .6138 & .4059 & & & \\ .1980 & .3960 & .1980 & .9900 & & \\ .3960 & .9900 & .4950 & .0000 & .9900 & \\ .1980 & .4950 & .2475 & .0000 & .4950 & .3465 \end{bmatrix}$$

$$S_b = \begin{bmatrix} 40.40 & & & & & \\ 58.58 & 125.24 & & & & \\ 29.29 & 62.62 & 41.41 & & & \\ 40.40 & 58.58 & 29.29 & 101.00 & & \\ 58.58 & 125.24 & 62.62 & 0.00 & 101.00 & \\ 29.29 & 62.62 & 41.41 & 0.00 & 50.50 & 35.35 \end{bmatrix}$$

Figure 4. Values of S and S_b for Artificial Data Example II.

Table 6

Estimated Values for Parameters Treated as Not Fixed in Example II

Parameter	Estimate	Asymptotic Error Variance	Parameter	Estimate	Asymptotic Error Variance
Λ_{31}	.500	<.001	$\Lambda_{b_{31}}$	.500	<.001
A_{21}	-.500	<.001	$A_{b_{21}}$	-.500	<.001
B_{11}	.200	<.001	$B_{b_{11}}$	.200	<.001
B_{12}	.400	<.001	$B_{b_{12}}$	.400	<.001
B_{21}	.300	<.001	$B_{b_{21}}$	.300	<.001
B_{22}	.800	<.001	$B_{b_{22}}$	.800	<.001
Γ_{31}	1.000	<.001	$\Gamma_{b_{31}}$	1.000	<.001
Σ_{ζ}	1.000	<.001	$\Sigma_{\zeta_{b_{11}}}$	1.000	<.001
Σ_{ζ_2}	.200	<.001	$\Sigma_{\zeta_{b_{22}}}$	.200	<.001
$\Sigma_{\theta_{11}}$	.030	<.001	$\Sigma_{\theta_{b_{11}}}$	.030	<.001
$\Sigma_{\theta_{22}}$	.100	<.001	$\Sigma_{\theta_{b_{22}}}$	.100	<.001
$\psi_{\varepsilon_{33}}$	.500	<.001	$\psi_{\varepsilon_{b_{33}}}$	.500	<.001
$\psi_{\omega_{33}}$	.100	<.001	$\psi_{\omega_{b_{33}}}$	.100	<.001

at least the same level of accuracy, yielding a value for the chi-square test of fit of 0.00 with 16 degrees of freedom.

The results of the analyses carried out thus far indicated that the estimation routine provides maximum likelihood estimates for parameters in all components of the model considered here. In addition, chi-square statistics for the test of fit of the model agreed with those independently arrived at by Schmidt for the cases where his data were reanalyzed. The chi-square value resulting from estimating parameters in the correctly specified model for the new set of artificial data was, as would be expected, quite close to zero. The asymptotic standard errors, while no strictly comparable values were available, appeared to generally agree with their approximations in Schmidt's work in instances where comparisons could reasonably be made. In the analysis of the new data these values were, as expected, quite close to zero. Based on these results, it was concluded that the estimation routine performed satisfactorily and could be used in conjunction with a real set of data. It is to the results of this effort that we now turn.

Analysis of Data Drawn from the National Longitudinal Study of the High School Class of 1972

As an attempt to illustrate the applicability of the model developed herein it was applied to a real set of data drawn from the National Longitudinal Study of the High School Class of 1972. Sponsored by the National Center for Education Statistics, the NLS is an ongoing large-scale survey project whose primary purpose is the observation of the educational and vocational activities, plans, aspirations, and attitudes of young people after they leave high school. The Educational Testing

Service began full-scale base year data collection in the spring of 1972. Data from over 18,000 seniors from a national probability sample of more than 1,000 high schools was collected. Beginning in the fall of the following year, Research Triangle Institute initiated the first of four follow-up surveys of these same subjects. As of the end of the third follow-up, over 13,000 subjects had responded to all of the instruments administered.

The particular variables drawn from this data base included sex, ethnicity, father's educational level, mother's educational level, hours of English and foreign language coursework, and reading and vocabulary scores. To facilitate the analysis, sex and ethnicity were recoded to binary-valued variables. For sex, zero represented female and one represented male; for ethnicity, zero represented Black and one represented white. Within- and between-school variance-covariance matrices were obtained involving these variables. These matrices are found in Figures 5 and 6 respectively.

The model to be estimated treated sex, ethnicity, father's educational level, and mother's educational level as observed exogenous variables. Hours of English and foreign language, together with reading and vocabulary scores, constituted the observed endogenous variables. Father's and mother's educational levels provided observed measures of a latent variable of socio-educational status. The hours of English and foreign language were seen as observed measures of a verbal-skill coursework variable. Likewise, reading and vocabulary scores were construed as measures of a verbal achievement trait.

The latent exogenous variables of sex, ethnicity, and socio-educational status were hypothesized to have a causal relationship with both

1	6,647.950							
2	42.695	22.305						
3	38.929	11.669	14.615					
4	1,273.014	261.267	215.373	24,678.005				
5	-.136	-.005	-.032	-7.345	.239			
6	1.043	.338	.250	3.303	.003	.099		
7	11.183	2.042	1.798	61.936	.015	.082	3.935	
8	9.318	1.563	1.403	47.324	.019	.054	1.653	2.624
	1	2	3	4	5	6	7	8

Variable	Name
1	Hours of English
2	Reading Score
3	Vocabulary Score
4	Hours of Foreign Language
5	Sex
6	Ethnicity
7	Father's Level of Education
8	Mother's Level of Education

Figure 5. Lower Triangular Elements of the Observed Within-School Variance-Covariance Matrix from NLS Data.

1	127,347.900							
2	-233.232	65.025						
3	-294.020	45.653	48.806					
4	1,789.193	1,172.148	1,155.997	123,295.380				
5	8.470	-.041	.073	-29.849	.440			
6	-5.683	4.211	3.474	8.340	.054	.909		
7	-98.283	18.654	17.521	628.049	.106	1.219	16.206	
8	-39.123	12.666	11.256	401.771	.138	.804	9.414	8.412
	1	2	3	4	5	6	7	8

Variable	Name
1	Hours of English
2	Reading Score
3	Vocabulary Score
4	Hours of Foreign Language
5	Sex
6	Ethnicity
7	Father's Level of Education
8	Mother's Level of Education

Figure 6. Lower Triangular Elements of the Observed Between-School Variance-Covariance Matrix from NLS Data.

verbal-skill coursework and verbal aptitude. Similar models were assumed to operate at both the within-and between-schools levels. The non-error-related components of the model are diagramatically presented in Figure 7, while the general parameterizations of the components of the variance-covariance matrices are set forth in Figure 8.

These variance-covariance matrices, together with the within- and between-group sample sizes, served as the input to the estimation routine. In spite of the expectations that a solution would be readily produced, even after 500 iterations the values of the derivatives of the non-fixed parameters had not converged on the criteria for the termination of the iterative estimation procedure. Examination of the intermediate estimates and the values of their first derivatives indicated that, while changes of considerable magnitude continued to take place at each iteration with respect to the parameter estimates, little improvement could be discerned in terms of their derivatives approaching zero.

Analysis of a Final Set of Artificial Data

As a final step in confirming that the problems experienced in estimating parameters for the model employing the NLS data were not simply due to some undetected flaw in the estimation routine, one additional set of artificial data was generated. This data was based upon a model of similar complexity as that used to produce the second artificial data set. The most pronounced difference lay in the fact that the elements of Σ_ζ and Σ_θ were no longer restricted to be relatively similar in magnitude. The values of the parameter matrices used to generate this data set are presented in Figure 9. Once again the same underlying structure prevailed at both the within- and between-groups level.

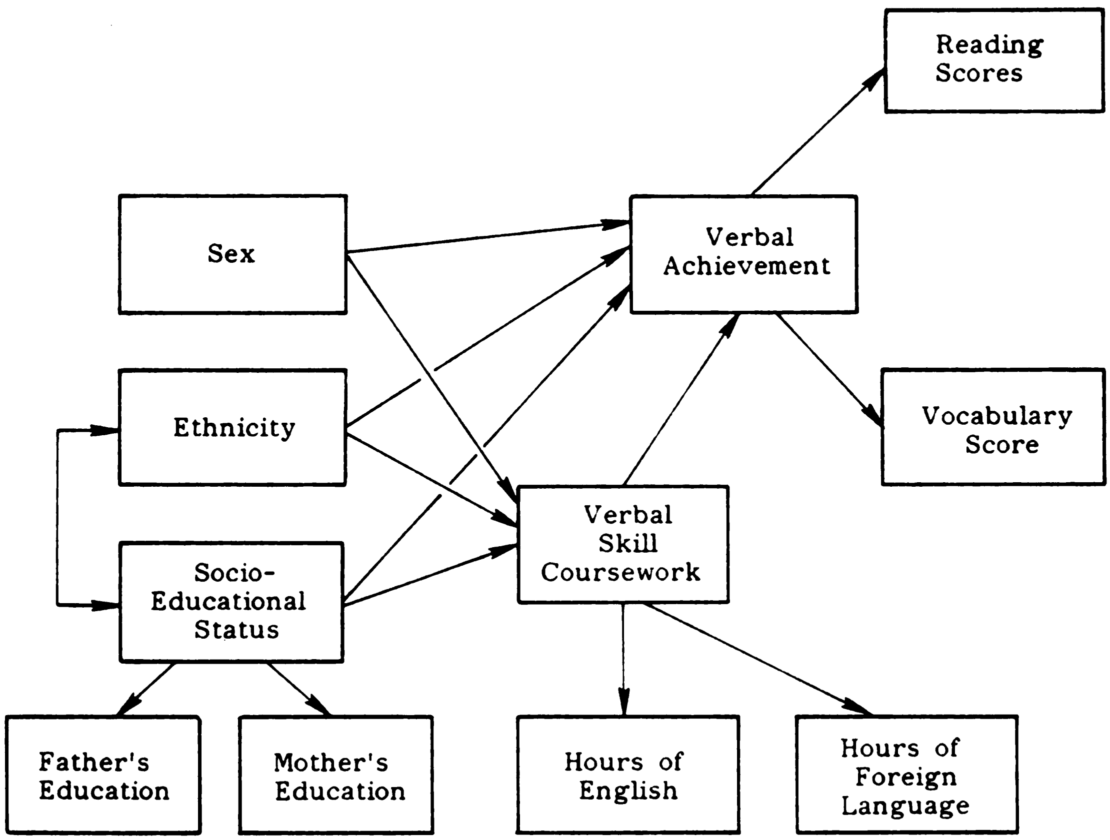

Figure 7. General Diagrammatic Structure of Estimated Model Using NLS Data.

$$\Sigma_{xx} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & \theta_1 \end{bmatrix}}_{\Gamma} \underbrace{\begin{bmatrix} \theta_2^* & 0 & 0 \\ 0 & \theta_3^* & \theta_4^* \\ 0 & \theta_4^* & \theta_5^* \end{bmatrix}}_{\Sigma_\zeta} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \theta_1 \end{bmatrix}}_{\Gamma'} + \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \theta_6 & 0 \\ 0 & 0 & 0 & \theta_7 \end{bmatrix}}_{\Psi_\omega}$$

$$\Sigma_{xy} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & \theta_1 \end{bmatrix}}_{\Gamma} \underbrace{\begin{bmatrix} \theta_2^* & 0 & 0 \\ 0 & \theta_3^* & \theta_4^* \\ 0 & \theta_4^* & \theta_5^* \end{bmatrix}}_{\Sigma_\zeta} \underbrace{\begin{bmatrix} \theta_8 & \theta_{11} \\ \theta_9 & \theta_{12} \\ \theta_{10} & \theta_{13} \end{bmatrix}}_{B'} \underbrace{\begin{bmatrix} 1 & 0 \\ \theta_{14} & 1 \end{bmatrix}^{-1}}_{(I-\Lambda)^{-1}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & \theta_{16} \\ 0 & 1 & \theta_{15} & 0 \end{bmatrix}}_{\Lambda'}$$

$$\Sigma_{yy} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & \theta_{15} \\ \theta_{16} & 0 \end{bmatrix}}_{\Lambda} \underbrace{\begin{bmatrix} 1 & 0 \\ \theta_{14} & 1 \end{bmatrix}^{-1}}_{(I-\Lambda)^{-1}} \left[\underbrace{\begin{bmatrix} \theta_8 & \theta_9 & \theta_{10} \\ \theta_{11} & \theta_{12} & \theta_{13} \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} \theta_2^* & 0 & 0 \\ 0 & \theta_3^* & \theta_4^* \\ 0 & \theta_4^* & \theta_5^* \end{bmatrix}}_{\Sigma_\zeta} \underbrace{\begin{bmatrix} \theta_8 & \theta_{11} \\ \theta_9 & \theta_{12} \\ \theta_{10} & \theta_{13} \end{bmatrix}}_{B'} + \underbrace{\begin{bmatrix} \theta_{17} & 0 \\ 0 & \theta_{18} \end{bmatrix}}_{\Sigma_0} \right] \underbrace{\begin{bmatrix} 1 & 0 \\ \theta_{14} & 1 \end{bmatrix}^{-1}}_{(I-\Lambda)^{-1}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & \theta_{16} \\ 0 & 1 & \theta_{15} & 0 \end{bmatrix}}_{\Lambda'} + \underbrace{\begin{bmatrix} \theta_{19} & 0 & 0 & 0 \\ 0 & \theta_{20} & 0 & 0 \\ 0 & 0 & \theta_{21} & 0 \\ 0 & 0 & 0 & \theta_{22} \end{bmatrix}}_{\Psi_\varepsilon}$$

* These parameters were constrained equal to their unrestricted maximum likelihood estimates at the between-groups level of the model to enhance the properties of the estimation routine.

Figure 8. General Parameterization of Model Variance-Covariance Matrices

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & .4 \end{bmatrix}$$

Λ & Λ_b

$$\begin{bmatrix} 1 & 0 \\ 5 & 1 \end{bmatrix}$$

(I-A) & (I-A_b)

$$\begin{bmatrix} 5 & .2 \\ .4 & 10 \end{bmatrix}$$

B & B_b

$$\begin{bmatrix} 10 & 1 \\ 0 & 100 \end{bmatrix}$$

Σ_ζ & Σ_{ζ_b}

$$\begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix}$$

Σ_θ & Σ_{θ_b}

$$\begin{bmatrix} 25.000 & 0.000 & 0.000 \\ 0.000 & 1,413.400 & 0.000 \\ 0.000 & 0.000 & 226.144 \end{bmatrix}$$

Ψ_ε & Ψ_{ε_b}

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ .5 & 0 \\ 0 & .5 \end{bmatrix}$$

Γ & Γ_b

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & .5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

Ψ_ω & Ψ_{ω_b}

Figure 9. Parameter Values Used to Generate Variance-Covariance Matrices for Example IV.

The resulting values for Σ and Σ_b are displayed in Figure 10. Based on values for n and m of 100, the final S and S_b matrices were derived and are set forth in Figure 11.

These data were then analyzed treating as free the parameters included in Figure 12. As with the NLS data, the estimation routine failed to provide stable parameter estimates even after more than 250 Fletcher-Powell iterations. In addition, correct values for the parameters were used as starting points for the estimation procedure. The first derivatives of the parameters evaluated at this point proved to be quite close to zero, as would be the case assuming the formulae were correct. Since the specific parameters being estimated and the form of the model itself is quite similar to that successfully treated in the case of the analyses of the second artificial data set, the problem is not with the program itself, which has successfully implemented the steepest descent and Fletcher-Powell methods. Further, the problem seems to be with their application to data sets which are difficult to analyze.

Attempted Solutions for the Estimation Problem

In analyzing the second set of artificial data for which parameters at both levels were defined to be equal, it was noted that, given identical starting values, the derivatives associated with the parameters at the within-groups level were larger than those for the comparable parameters at the between-groups level by a factor associated with the number of subjects within each group. Since the steepest descent method alters estimated values for parameters in proportion to the size of their first derivatives, changes in estimated parameter value first took place with respect to the within-groups parameters. Initial values

$$\Sigma = \Sigma_b = \begin{bmatrix} 280 & & & & & & \\ 1,495 & 20,000 & & & & & \\ 598 & 7,434.64 & 3,200 & & & & \\ 50 & 254 & 101.6 & 11 & & & \\ 20 & 1,100 & 440 & 0 & 110 & & \\ 25 & 127 & 50.8 & 5 & 0 & 3 & \\ 10 & 550 & 220 & 0 & 50 & 0 & 30 \end{bmatrix}$$

Figure 10. Values of Σ and Σ_b Matrices for Example IV.

$$S = \begin{bmatrix} 277.20 & & & & & & \\ 1,480.05 & 19,800.00 & & & & & \\ 592.02 & 7,360.29 & 3,168.00 & & & & \\ 49.50 & 251.46 & 100.58 & 10.89 & & & \\ 19.80 & 1,089.00 & 435.60 & 0.00 & 108.90 & & \\ 24.75 & 125.73 & 50.29 & 4.95 & 0.00 & 2.97 & \\ 9.90 & 544.50 & 217.80 & 0.00 & 49.50 & 0.00 & 29.70 \end{bmatrix}$$

$$S_b = \begin{bmatrix} 28,280.00 & & & & & & \\ 150,995.00 & 2,020,000.00 & & & & & \\ 60,398.00 & 750,898.64 & 323,200.00 & & & & \\ 5,050.00 & 25,654.00 & 10,261.60 & 1,111.00 & & & \\ 2,020.00 & 111,100.00 & 44,440.00 & 0.00 & 11,110.00 & & \\ 2,525.00 & 12,827.00 & 5,130.80 & 505.00 & 0.00 & 303.00 & \\ 1,010.00 & 55,550.00 & 22,220.00 & 0.00 & 5,050.00 & 0.00 & 3,030.00 \end{bmatrix}$$

Figure 11. Lower Triangular Elements of the S and S_b Matrices for Example IV.

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & \theta_1 \end{bmatrix}$$

Λ & Λ_b

$$\begin{bmatrix} 1 & 0 \\ \theta_2 & 1 \end{bmatrix}$$

(I-A) & (I-A_b)

$$\begin{bmatrix} \theta_3 & \theta_4 \\ \theta_5 & \theta_6 \end{bmatrix}$$

B & B_b

$$\begin{bmatrix} \theta_7 & 1 \\ 0 & \theta_8 \end{bmatrix}$$

Σ_ζ & Σ_{ζ_b}

$$\begin{bmatrix} \theta_9 & 0 \\ 0 & \theta_{10} \end{bmatrix}$$

Σ_θ & Σ_{θ_b}

$$\begin{bmatrix} 25.000 & 0.000 & 0.000 \\ 0.000 & 1,413.400 & 0.000 \\ 0.000 & 0.000 & \theta_{11} \end{bmatrix}$$

Ψ_ε & Ψ_{ε_b}

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \theta_{12} & 0 \\ 0 & \theta_{13} \end{bmatrix}$$

Γ & Γ_b

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & .5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

Ψ_ω & Ψ_{ω_b}

Figure 12. General Model Underlying Example IV.

for the Fletcher-Powell algorithm were, as a result, closer to the actual value for the within-groups parameters than for those for the between-groups parameters.

Initial efforts of the Fletcher-Powell algorithm were also directed toward changes in the values of the within-group parameters since Fletcher-Powell departs from steepest descent only as the successive corrections to the initial estimate of the information matrix (an identity matrix) make themselves felt. As a result, the values of the between-groups parameters tended to lag behind those of their corresponding within-groups parameters. Considerable oscillation in their values continued to be observed even after the values of the within-groups parameters had substantially stabilized.

One set of attempts to improve the behavior of the estimation procedure involved forcing the steepest descent segment to perform additional iterations. In retrospect, it was not terribly surprising that the marginal improvements in reducing the number of Fletcher-Powell iterations for the simpler problems did not translate into a successful approach to solving the problems associated with the more difficult problems.

As an alternative, the steepest descent segment was modified so as to alternate between activity in changing the values of all parameters and simply concentrating on changes in the values of the between-groups parameters holding the values of the within-groups parameters fixed. When this modified approach was tried with the data sets that had demonstrated convergence previously, small reductions in the number of Fletcher-Powell iterations for the estimation procedure to arrive at solutions were observed. Once again, the modified estimation procedure failed to generate correct, fully-converged estimates for the parameters associated with the more difficult data.

When the third set of artificial data was used as input to the estimation routine a potentially revealing result was obtained. Once again, the derivatives displayed the same pattern (vis à vis) the different levels of the parameters. In addition, however, several specific parameters were observed to have values for their derivatives that far exceeded those of the other parameters at the same level. The steepest descent phase terminated after making some modifications to the values of these parameters and little impact on those associated with the remaining parameters. Fletcher-Powell proceeded in the same vein until from 30 to 50 iterations had taken place. At that time, considerable changes were observed to take place in the values of nearly all parameters. This would appear to reflect the attainment of an approximate information matrix that departed substantially from the identity matrix.

Despite the substantial changes in the values of all the parameters, convergence proved to be elusive. Many of the parameters were observed to take on relatively stable values while several continued to slowly oscillate. Inspection of the relatively stable parameter values revealed them to be within 10 or 20 percent of the generating values. Those for the unstable parameters proved to be nearly unrelated to those of their progenitors.

An Illustrative Interpretation of the NLS Results

At this point it may be useful to illustrate how interpretation of the results from an application of the model might be performed. While the estimation procedure failed to yield fully converged estimates for the parameters in the model dealing with the NLS data, the intermediate values at the point at which the estimation routine stopped provide a reasonable set of results for this purpose. These intermediate results are presented in Table 7. The following interpretation of these results is based upon the assumptions that the model had acceptable fit to the data and that all of the point estimates differed from zero by more than two standard errors.

For ease in interpreting the parameter estimates associated with the linkages in the model (with the exception of those associated with measurement error), the estimates have been included in the diagrammatic form of the model presented in Figure 13. The estimates arising at the within-schools level appear alone while those at the between-schools level are contained within parentheses. Parameter values that were fixed are underlined to distinguish them from those which were unconstrained during the estimation process.

With respect to both the between- and within-schools levels there are two general sets of results that might be of interest. The first involves the measurement aspect of the model while the second is associated with the interrelationships among the latent variables themselves. To keep the discussion at a more substantive level, the measurement related results are not addressed in any great detail except to note that the very large values associated with θ_{19} and θ_{22} at both levels points to a serious problem in the definition of a common, verbal coursework variable. This is clearly a function of the low degree of association between the two variables. Were the model to be reformulated based on these results, it would be preferable to posit independent latent variables

Table 7

Intermediate Parameter Estimates for NLS Data

General Parameter	Within-School Level Estimate	Between-School Level Estimate
θ_1	.762	.643
θ_2	.259	.014
θ_3	.108	.062
θ_4	.084	.093
θ_5	2.352	.919
θ_6	1.910	.008
θ_7	1.590	.051
θ_8	-5.042	-.775
θ_9	2.323	-.378
θ_{10}	4.680	.304
θ_{11}	.868	-.633
θ_{12}	2.053	3.267
θ_{13}	.085	.908
θ_{14}	.209	.809
θ_{15}	.823	.903
θ_{16}	6.107	179.138
θ_{17}	166.412	.152
θ_{18}	5.155	.379
θ_{19}	6973.252	9284.864
θ_{20}	8.803	.343

(continued)

Table 7 (continued)

General Parameter	Within-School Level Estimate	Between-School Level Estimate
θ_{21}	5.429	.234
θ_{22}	18297.432	.002

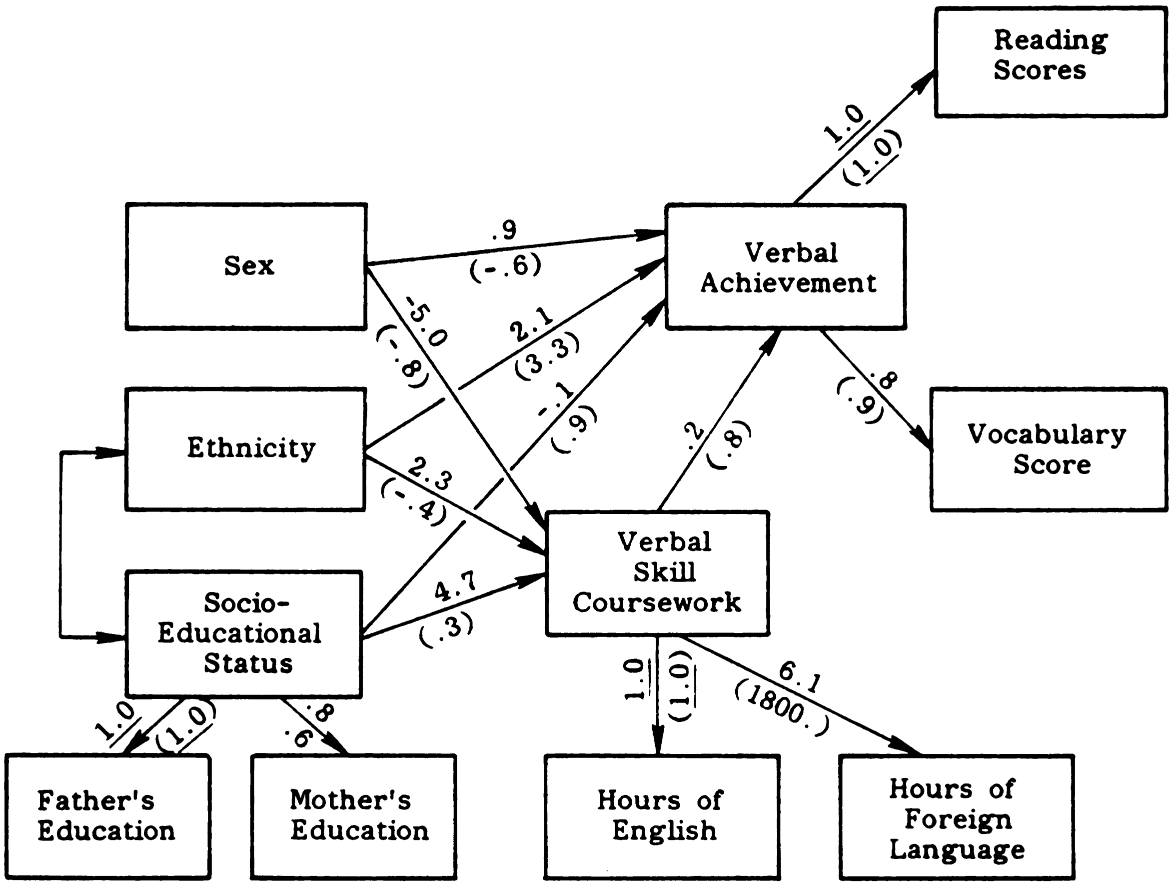

Figure 13. General Diagrammatic Structure of Estimated Model Using NLS Data with Selected Parameter Estimates.

associated with both of these variables rather than one common latent variable.

Turning to the estimated relationships among the latent exogeneous and indogeneous variables set forth in Figure 13 we now consider the within- and between-schools results in turn. At the within-schools level several interesting results may be noted. Sex, ethnicity, and socio-educational status (SEdS) are all related to verbal skill coursework in ways that might be expected. The positive coefficients for ethnicity and SEdS simply indicate that, at the within-schools level, those who are white and those who come from families with parents having high levels of education tend to take more units of verbal-related courses. The negative coefficient associated with sex indicates that males are less inclined to take such courses, all else being equal.

With respect to verbal achievement the results at the within-schools level are not entirely anticipated. Sex has a coefficient with a positive value while SEdS has a negative, albeit small, value. As would be expected, ethnicity and verbal skill coursework both are related to verbal achievement with positive coefficients. It would appear from these results that the typical univariate relationships observed among sex, parents' education, and verbal achievement are explained more by the indirect effects of these two background variables through verbal skill coursework than through direct effects on verbal achievement itself.

The results at the between-schools level differ considerably in both magnitude and direction from those at the within-schools level. The coefficients linking the school-level aggregates of sex, ethnicity, and SEdS to verbal skill coursework are considerably smaller than their counterparts at the within-schools level. In addition, at this level, ethnicity is associated with verbal skills coursework through a coefficient having a negative value implying that, for schools of equivalent sex and SEdS, those that are not composed entirely of white students have marginally more verbal skill coursework taken by the students.

With respect to the coefficients relating sex, ethnicity, SEdS, and verbal skill coursework to verbal achievement at the between-schools level several differences may also be noted with the results at the within-schools level. At this level, sex has a small negative coefficient while the SEdS variable has a coefficient with a small positive value. Both are opposite in sign to their within-schools counterparts. On the other hand, both ethnicity and verbal skill coursework are associated with positive coefficients as they are at the within-schools level.

In general the exogeneous school-level aggregate variables of sex, ethnicity and SEdS have a much weaker effect on verbal skills coursework at this level than at the within-schools level. This would appear to reflect an institutional emphasis on verbal skill type coursework that is considerably less sensitive to such factors as sex, ethnicity, and parents' educational level than the behavior of the students within the schools. Unfortunately, such factors as ethnicity, parents' level of education, and verbal skill coursework are even more strongly related to verbal achievement at the between-schools level than at the within-schools level.

The results discussed in this section should be viewed as illustrative of the type of interpretation afforded through the use of this model. Since the parameter estimates used were not fully-converged estimates, their values should not be used to come to any real substantive conclusions with repsect to this set of variables. Furthermore, the lack of any estimated standard errors makes the exercise carried out at this point even more tentative in nature.

Chapter 6

Summary of Results, Conclusions, and New Directions

In some respects the nature of the work presented here is atypical of that necessary for most dissertations in educational psychology. Rather than being directed at answering a specific set of questions through the use of available analytic techniques, its purpose was the implementation of a relatively new analytic technique in the context of multi-level data. As such, the most desirable result would be a useful process.

The results of the work carried out thus far are not, therefore, confined simply to the set of analyses performed, but include the development of the components necessary for those analyses. These components included the statement of the model itself, the first and second derivatives of the effective part of the log likelihood function and the computer program which makes use of them for the estimation of parameter values and asymptotic standard errors. Inasmuch as can be determined from the behavior of the computer programs on the first two sets of artificial data where the expected and correct results were obtained, the process has been defined and implemented. The question that remains at this point has to do with its broader usefulness. The failure of the estimation routine when applied to the NLS data clearly indicates that, as things now stand, the process cannot be successfully implemented for all sets of data. In the following section, these results are reviewed and the implications for further work in this area are considered. For the convenience of the reader, where equations are referenced they appear fully, with their original equation number.

<u>The Model</u>

The model which was developed has, as its basis, the simple random effects model in the multivariate form. Thus, the overall variance-covariance matrix was seen as being composed of two additive components:

$$\Sigma_z = \Sigma + \Sigma_b \tag{34}$$

where the terms on the right hand side of the equation arise at the between-groups and within-groups levels, respectively.

Under the assumption of multivariate normality, previous authors have addressed the problem of arriving at unrestricted maximum likelihood estimates for Σ_b and Σ. Work has also been carried out to permit the restricted maximum likelihood estimation under the constraints of some very simple models. The efforts of the current author were directed at formulating a more general structural equation model applicable to this type of hierarchical data. The model developed is applicable to both the within-groups and between-groups variance-covariance matrices simultaneously, and was generally patterned after the linear structural equation model considered by Jöreskog, Wiley, and others.

The full models that were developed for Σ_b and Σ are as follows:

$$\Sigma = \left[\begin{array}{c|c} \Gamma\Sigma_\zeta\Gamma' + \Psi_\omega & \Gamma\Sigma_\zeta B'(I-A)^{-t}\Lambda' \\ \hline (I-A)^{-1}B\Sigma_\zeta\Gamma' & \left[\begin{array}{c} \Lambda[(I-A)^{-1}B\Sigma_\zeta B'(I-A)^{-t}]\Lambda' \\ + \\ \Lambda[(I-A)^{-1}\Sigma_\theta(I-A)^{-t}]\Lambda' + \Psi_\varepsilon \end{array}\right] \end{array}\right] \tag{35}$$

$$\Sigma_b = \left[\begin{array}{c|c} \Gamma_b \Sigma_{\zeta_b} \Gamma_b' + \Psi_{\omega_b} & \Gamma_b \Sigma_{\zeta_b} B_b'(I-A_b)^{-t} \Lambda_b' \\ \hline \Lambda_b (I-A_b)^{-1} B_b \Sigma_{\zeta_b} \Gamma_b' & \left[\begin{array}{c} \Lambda_b [(I-A_b)^{-1} B_b \Sigma_{\zeta_b} B_b' (I-A_b)^{-t}] \Lambda_b' \\ + \\ \Lambda_b [(I-A_b)^{-1} \Sigma_{\theta_b} (I-A_b)^{-t}] \Lambda_b' + \Psi_{\varepsilon_b} \end{array}\right] \end{array}\right] \quad (36)$$

Maximum Likelihood Estimation

Under the assumption that the underlying data follow a multivariate normal distribution, Schmidt (1969) has shown that the effective part of the log likelihood function, where the parameters are simply Σ and Σ_b, can be expressed as follows:

$$\log L = \frac{m-mn}{2} \log (|\Sigma|) - \frac{m}{2} \log (|\Sigma + n\Sigma_b|) - \frac{mn}{2} \operatorname{tr} \{\Sigma^{-1} S\} - \frac{m}{2} \operatorname{tr} \{(\Sigma + n\Sigma_b)^{-1} S_b\}. \quad (20)$$

where

$$S = \frac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} (\underline{y}_{ij} - \underline{y}_{i.})(\underline{y}_{ij} - \underline{y}_{i.})' \quad (17)$$

$$S_b = \frac{n}{m} \sum_{i=1}^{m} (\underline{y}_{i.} - \underline{y}_{..})(\underline{y}_{i.} - \underline{y}_{..})' \quad (18)$$

Substitution of the parametric expressions for Σ and Σ_b in this equation yielded the fully paramterized version of the log likelihood function. The values of the parameters which maximize this function for a given S and S_b are maximum likelihood estimates of the parameters.

Test of Fit and Standard Error Estimation

Given a set of maximum likelihood estimates for the parameter matrices in a particular application, it was seen possible to produce a statistical test of the fit of the model to a given set of data. This test made use of the ratio of the value of the likelihood function evaluated at the solution point for the maximum likelihood estimates to the value over the unrestricted solution space. The test statistic,

$$\chi^2 = 2(\log L_{\text{unrestricted}} - \log L_{\text{restricted}}), \tag{50}$$

is, for large sample sizes, distributed as chi-square with degrees of freedom equal to the difference between the number of unique elements in Σ_b and Σ and the number of unique parameters estimated in the model.

It was also possible to produce asymptotic standard errors associated with the estimated parameters. The procedure considered made use of the fact that the asymptotic covariance matrix of the maximum likelihood estimators is equal to the negative inverse of the expected value of the matrix of second partial derivatives. Based on earlier work by Schmidt (1969) it was seen that this could be expressed as follows:

$$E\left[\frac{\partial^2 \log \ell}{\partial\theta_i \partial\theta_j}\right] = -\frac{1}{2}\,\mathrm{tr}\left\{\frac{\partial^2(\Sigma + n\Sigma_b)}{\partial\theta_i\partial\theta_j}(\Sigma + n\Sigma_b)^{-1}\right\} \tag{48}$$

$$+\ \frac{m(1-n)}{2}\,\mathrm{tr}\left\{(\Sigma + n\,\Sigma_b)^{-1}\frac{\partial(\Sigma + n\,\Sigma_b)}{\partial\theta_i}(\Sigma + n\,\Sigma_b)^{-1}\frac{\partial(\Sigma + n\,\Sigma_b)}{\partial\theta_j}\right\}.$$

Thus, it was necessary to obtain expressions for the first and second derivatives of Σ and Σ_b with respect to individual parameters and parameter pairs, respectively. These are set forth fully in Appendices B and C.

Obtaining the Maximum Likelihood Estimates

It was seen that obtaining the maximum likelihood estimates of the parameters in a particular application is not a simple process. In theory, the solutions for the set of equations resulting from setting the first partial derivatives of the log likelihood function equal to zero would provide estimating formulae for the various parameter estimates. The complexity of the set of simultaneous equations precluded the derivation of such a set of formulae. Alternatively, a set of numerical procedures based on the method of steepest descent and the method of Davidon-Fletcher-Powell were adopted to provide the values of the estimates for any particular application. The matrix expressions for the first partial derivatives of the log likelihood function, necessary for the application of both methods, were derived and presented in Appendix A. The adequacy of these approaches is considered in the discussion of the results of their application.

Results of Analyses

The first set of analyses which estimated parameters in the model made use of a set of data employed by Schmidt (1969). As a result of Schmidt's work, the parameter estimates for four relatively simple models of the sort considered herein were already known. When the present estimation procedure was applied to those data to estimate the parameters in the same model the same results were obtained in each case. It was concluded that the estimation procedures implemented by the present author were correct and satisfactory insofar as these simple models were concerned.

A new set of data was generated based on a more complex structure than that found in Schmidt's work. The model used to generate this second set of data contained non-zero values for at least one element in each parameter matrix of the full model. This set of data was then analyzed employing the correctly specified model to see if the generating values would be faithfully reproduced. Once again, the estimation procedure performed adequately and yielded the expected results.

Since the estimation procedure would be highly unlikely to provide correct parameter estimates in the event that some error were present anywhere in the conceptual process, including the programming stage, it seemed reasonable to conclude that the estimation procedures had been, in fact, successfully implemented.

The estimation routine was then used in an attempt to generate parameter estimates for a model which addressed aspects of the within- and between-school variability of a set of variables drawn from the National Longitudinal Study of the High School Class of 1972. Despite the use of an excessive number of iterations, the estimation routine failed to yield fully converged estimates for the free parameters in the hypothetical model.

In an attempt to gain a better understanding of the failure of the estimation procedure in the NLS application, an additional set of artificial data were generated using a model which would yield sharply unequal variance terms within each variance-covariance matrix. When the estimation routine was applied to this set of data using the correctly specified model, a satisfactory solution was not obtained.

A variety of attempts were then carried out to improve the convergence of the estimation procedure. Initially, the procedure was modified to require the performance of a larger number of steepest descent iterations. Further modifications were directed at allowing the steepest descent method to alternate between improvements on all parameters and improvements on the between-groups parameters only. These efforts failed to produce an estimation procedure that would accurately replicate the generating parameters associated with the final artificial data set.

Conclusions

At this point, it seems safe to conclude that the original goals of the effort reported here have been met to some extent. The model for linear structural relations applicable to multi-level data was successfully derived. The conditions that must be satisfied for the attainment of maximum likelihood estimates of the parameters in the model were derived. Procedures for testing the fit of an estimated model and for producing asymptotic standard errors associated with estimated parameters were set forth. Finally, a computer program intended to yield parameter estimates through the use of iterative methods was successfully implemented.

What remains as a problem confronting the general use of this set of products is the inability of the estimation routine to provide satisfactory parameter estimates when confronted with a difficult set of data. Should it have been possible to specify the conditions under which convergence could be assured, this handicap would not prove to be such a problem. Alternatively, had an alternative estimation procedure insensitive to such problems been found, this problem would have been overcome. Such was not the case. Further work in either or both of thes areas is clearly necessary.

While the constraints facing the present author precluded an attack on either of these fronts, experience with the currently implemented estimation routine provided some results leading to speculation on the direction in which efforts to attain the latter goal should proceed. This speculation is briefly set out below.

For the steepest descent procedure to operate effectively in making progress toward the maximum likelihood estimates associated with a particular problem it seems necessary that the matrix of second derivatives be similar, within a constant multiple, to an identity matrix. In this situation, changes in the intermediate values of the parameter estimates would proceed relatively uniformly. The nature of the current application vitiates against this. Where the structure at both levels of the hierarchy is the same, changes take place at the within-groups level much more rapidly than at the between-groups level. In light of the behavior of the estimation procedure when applied to the third set of artificial data, it seems clear that this phenomenon can also take place independently of the multi-level nature of the data to which it applies with some parameters at a given level being subject to substantial changes while others change but little.

It is also clear that the effective operation of the Davidon-Fletcher-Powell procedure is dependent upon the characteristics of the data with which it operates. Jöreskog has indicated the the intermediate parameter estimates associated with the LISREL problem must be "close" to the solution point for convergence to take place. The nature of this closeness would seem to be related to obtaining a region within which the matrix of second derivatives is relatively constant. Ideally, steepest descent would terminate only after taking the intermediate estimates into such a region. In the current application, it seems as though this does not happen. At least with the two more difficult sets of data to which the currently implemented estimation procedures have been applied, the steepest descent phase terminates prior to the attainment of such a region. At these points, the Fletcher-Powell routine is incapable of providing a converged solution, probably because the matrix of second derivatives is quite variable within these regions.

Should the foregoing speculation prove to characterize the nature of the estimation problem encountered in the course of the current research, any iterative approach which will generally be capable of the attainment of maximum likelihood estimates for the type of model addressed here needs to explicitly incorporate the matrix of second derivatives and not an iterative approximation to it. Thus, it seems that the most promising approach to adopt is that of Newton's method, which makes use of expressions for both the first and second derivatives of the likelihood function with respect to the parameters of the model.

LIST OF REFERENCES

LIST OF REFERENCES

Bielby, W. T., R. M. Hauser, D. L. Featherman. "Response Errors of Black and Nonblack Males in Models of the Intergenerational Transmission of Socioeconomic Status." American Journal of Sociology, 82 (1977), pp. 1242-1288.

Blalock, H. M., Jr. Causal Inference in Non-Experimental Research. Chapel Hill: University of North Carolina Press, 1964.

Blalock, H. M., Jr. "Multiple Indicators and the Causal Approach to Measurement Error." American Journal of Sociology, 75 (1969), pp. 264-272.

Blalock, H. M., Jr. Causal Models in the Social Sciences. Chicago: Aldine-Atherton, 1971.

Bock, R. D. "Components of Variance Analysis As a Structural and Discriminal Analysis for Psychological Tests." British Journal of Statistical Psychology, 13 (1960), pp. 151-163.

Bock, R. D. Multivariate Statistical Methods in Behavioral Research. New York: McGraw-Hill, 1965.

Bock, R. D. and R. E. Bargmann. "Analysis of Covariance Structure." Psychometrika, 31 (1966), pp. 507-534.

Brookover, Wilbur, Charles Beady, Patricia Flood, John Schweitzer, and Joe Wisenbaker. School Social Systems and Student Achievement. New York: Praeger, 1979.

Burstein, L. J. and M. D. Miller. "The Use of Within-Group Slopes as Indices of Group Outcomes." Paper presented at the meeting of the American Educational Research Association, San Francisco, 1976.

Burstein, L. J. "Assessing Classroom Effects Using Within Class Regression Curves." Paper presented at the meeting of the American Educational Research Association, San Francisco, 1979.

Burt, C. "Factor Analysis and Analysis of Variance." British Journal of Psychology, 1 (1947), pp. 3-26.

Cramer, J. S. Empirical Econometrics. Amsterdam: North Holland Publishing Company, 1969.

Creasey, M. A. "Analysis of Variance as an Alternative to Factor Analysis." Journal of the Royal Statistical Society, 19 (1957), pp. 318-325.

Cronbach, L. J. "Research on Classrooms and Schools: Formulation of Questions, Design and Analysis." An occasional paper of the Stanford Evaluation Consortium, undated.

Davidon, W. C. "Variance Algorithm for Minimization." Computer Journal, 10 (1968), pp. 406-410.

Duncan, O. D. "Path Analysis: Sociological Examples." American Journal of Sociology, 72 (1966), pp. 1-16.

Duncan, O. D. Introduction to Structural Equation Models. New York: Academic Press, 1975.

Fisher, F. M. The Identification Problem in Econometrics. New York: McGraw-Hill, 1966.

Fisher, F. M. "Generalization of the Rank and Order Condition for Identifiability." In Readings in Econometric Theory. Eds. J. M. Dowling and F. R. Glahe. Boulder: Colorado Associated University Press, 1970, pp. 331-347.

Fletcher, R. and M. J. D. Powell. "A Rapidly Convergent Descent Method for Minimization." Computer Journal, 6 (1963), pp. 163-168.

Geraci, V. J. "Identification of Structural Equation Models with Measurement Error." In Latent Variables in Socio-Economic Models. Eds. D. J. Aigner and A. S. Goldberger. Amsterdam: North Holland Publishing Company, 1977, pp. 163-185.

Glendening, Linda. "The Effects of Correlated Units of Analysis: Choosing the Appropriate Unit." Paper presented at the meeting of the American Educational Research Association, San Francisco, 1976.

Gruvaeus, G. T. and R. G. Jöreskog. A Computer Program for Minimizing a Function of Several Variables. Princeton, New Jersey: Educational Testing Service, 1970.

Hannan, M. T. "Aggregation Gain Reconsidered." Paper presented at the meeting of the American Educational Research Association, San Francisco, 1976.

Harman, H. H. Modern Factor Analysis. Chicago: University of Chicago Press, 1967.

Hemmerle, W. J. "Obtaining Maximum-Likelihood Estimates of Factor Loadings and Communalities Using an Easily Implemented Iterative Computer Procedure." Psychometrika, 30 (1965), pp. 291-301.

Jöreskog, Karl G. "Testing a Simple Structure Hypothesis in Factor Analysis." Psychometrika, 31 (1966), pp. 165-178.

Jöreskog, Karl G. UMFLA: A Computer Program for Unrestricted Maximum Likelihood Factor Analysis. Princeton, New Jersey: Educational Testing Service, 1966.

Jöreskog, Karl G. "Some Contributions to Maximum Likelihood Factor Analysis." Psychometrika, 32 (1967), pp. 443-482.

Jöreskog, Karl G. "A General Approach to Confirmatory Maximum Likelihood Factor Analysis." Psychometrika, 34 (1969), pp. 183-202.

Jöreskog, Karl G. "A General Method for Analysis of Covariance Structures." Biometrika, 57 (1970), pp. 239-251.

Jöreskog, Karl G. "Estimation and Testing of Simplex Models." British Journal of Mathematical and Statistical Psychology, 23 (1970), pp. 121-145.

Jöreskog, Karl G. "Statistical Analysis of Sets of Congeneric Tests." Psychometrika, 36 (1971), pp. 103-133.

Jöreskog, Karl G. and Marielle van Thillo. LISREL: A General Computer Program for Estimating a Linear Structural Equation System Involving Multiple Indicators of Unmeasured Variables. Princeton, New Jersey: Educational Testing Service, 1972.

Jöreskog, Karl G. "Analysis of Covariance Structures." In Multivariate Analysis III. Ed. P. R. Krishnaiah. New York: Academic Press, 1973, pp. 263-285.

Jöreskog, Karl G. "Analyzing Psychological Data by Structural Analysis of Covariance Matrices." In Contemporary Developments in Mathematical Psychology-Volume II. Eds. R. C. Atkinson, D. H. Krantz, R. D. Luce, and P. Suppes. San Francisco: W. H. Freeman and Company, 1974, pp. 1-56.

Jöreskog, Karl G. "Structural Equation Models in the Social Sciences: Specification, Estimation and Testing." In Applications of Statistics. Ed. P. R. Krishnaiah. Amsterdam: North Holland Publishing Company, 1977, pp. 265-287.

Jöreskog, Karl G. and A. S. Goldberger. "Estimation of a Model with Multiple Indicators and Multiple Causes of a Single Latent Variable." Journal of the American Statistical Association, 70 (1973), pp. 631-639.

Jöreskog, Karl G. and G. Gruvaeus. RMFLA: A Computer Program for Restricted Maximum Likelihood Factor Analysis. Princeton, New Jersey: Educational Testing Service, 1967.

Jöreskog, Karl G. and D. F. Lawley. "New Methods in Maximum Likelihood Factor Analysis." British Journal of Mathematical and Statistical Psychology, 21 (1968), pp. 85-96.

Kendall, M. G. and A. Stuart. The Advanced Theory of Statistics, Vol. II. New York: Hafner, 1961.

Kerlinger, F. Foundations of Behavioral Research. New York: Holt, Rinehart, and Winston, 1973.

Kirk, R. E. Experimental Design: Procedures for the Behavioral Sciences. Belmont: Wadsworth, 1968.

Kohn, M. L. and C. Schooler. "The Reciprocal Effects of the Substantive Complexity of Work and Intellectual Flexibility: A Longitudinal Assessment." American Journal of Sociology, 84 (1978), pp. 24-52.

Koopmans, T. C. "Identification Problems in Economic Model Construction." Econometrica, 17 (1949), pp. 128-143.

Koopmans, T. C. and O. Reiersøl. "The Identification of Structural Characteristics." Annals of Mathematical Statistics, 21 (1950), pp. 165-181.

Kuhn, T. S. The Structure of Scientific Revolutions. Chicago: The University of Chicago Press, 1970.

Lawley, D. N. "The Estimation of Factor Loadings by the Method of Maximum Likelihood." Proceedings of the Royal Society of Edinburgh, 40 (1940), pp. 64-82.

Lawley, D. N. "Estimation in Factor Analysis Under Various Initial Assumptions," British Journal of Statistical Psychology, 11 (1958), pp. 1-12.

Lawley, D. N. and A. E. Maxwell. Factor Analysis as a Statistical Method. London: Butterworth's, 1963.

Lawley, D. N. and A. E. Maxwell. Factor Analysis as a Statistical Method, 2nd Ed. New York: American Elsevier, 1971.

Lord, F. M. and M. E. Novick. Statistical Theories of Mental Test Scores. Reading: Addison-Wesley, 1968.

Luenberger, D. G. Introduction to Linear and Nonlinear Programming. Reading: Addison-Wesley, 1973.

Malinvaud, E. Statistical Methods of Econometrics. Chicago: Rand McNally, 1966.

Malinvaud, E. Statistical Methods of Econometrics. 2nd Revised Ed. Amsterdam: North Holland Publishing Company, 1970.

Maw, C. E. "The Problem of Data Aggregation and Cross-Level Inference with Categorical Data." Paper presented at the meeting of the American Educational Research Association, San Francisco, 1976.

McDonald, R. P. "Testing Pattern Hypotheses for Covariance Matrices." Psychometrika, 39 (1974), pp. 189-200.

McDonald, R. P. "Testing Pattern Hypotheses for Correlation Matrices." Psychometrika, 40 (1975), pp. 253-255.

McDonald, R. P. and H. Swaminathan. "A Simple Matrix Calculas with Applications to Structural Models for Multivariate Data--Part I: Theory." The Ontario Institute for Studies in Education, undated.

Miller, J., C. Schooler, M. L. Kohn, and K. A. Miller. "Women and Work: The Psychological Effects of Occupational Conditions." American Journal of Sociology, 85 (1979), pp. 66-94.

Morrison, D. F. Multivariate Statistical Methods. New York: McGraw-Hill, 1967.

Mortimer, J. T. and J. Lorence. "Work Experience and Occupational Value Socialization: A Longitudinal Study." American Journal of Sociology, 84 (1979), pp. 1361-1385.

Murkherjee, B. N. "Likelihood Ratio Tests of Statistical Hypotheses Associated with Patterned Covariance Matrices in Psychology." British Journal of Mathematical and Statistical Psychology, 23 (1970), p. 120.

Murkherjee, B. N. "Techniques of Covariance Structural Analysis." Australian Journal of Statistics, 18 (1976), pp. 131-150.

Potthoff, R. F. and S. N. Roy. "A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems." Biometrika, 51 (1964), pp. 313-316.

Rao, C. R. Advanced Statistical Methods in Biometric Research. New York: Hafner Press, 1952.

Rao, C. R. Linear Statistical Inference and Its Applications. New York: Wiley, 1965.

Rock, D. A., C. E. Werts, R. L. Linn, and R. G. Jöreskog. "A Maximum Likelihood Solution to the Errors in Variables and Errors in Equations Model." Journal of Multivariate Behavioral Research, 12 (1977), pp. 187-197.

Schmidt, W. H. Covariance Structure Analysis of the Multivariate Random Effects Model. Dissertation, University of Chicago, 1969.

Schmidt, W. H. "Structural Equation Models and Their Application to Longitudinal Data." Paper presented at the Conference on Longitudinal Statistical Analysis, Boston, 1975.

Schmidt, W. H. and D. E. Wiley. "Analytic Problems in Longitudinal Data." Paper presented at the Conference on Methodological Concerns in Evaluational Research, Chicago, 1974.

Tiao, G. C. and W. V. Tan. "Bayesian Analysis of Random Effect Models in the Analysis of Variance, I: Posterior Distributions of Variance Components." Biometrika, 52 (1965), pp. 37-53.

Tiao, G. C. and W. V. Tan. "Bayesian Analysis of Random Effect Models in the Analysis of Variance, II: The Effect of Auto-Correlated Errors." Biometrika, 53 (1966), pp. 477-495.

Wiley, D. E. "The Identification Problem for Structural Equation Models with Unmeasured Variables." In Structural Equation Models in the Social Sciences. Eds. A. S. Goldberger and O. D. Duncan. New York: Seminar Press, 1973, pp. 69-83.

Wiley, D. E., W. H. Schmidt, and W. J. Bramble. "Studies of a Class of Covariance Structure Models." Journal of the American Statistical Association, 68 (1973), pp. 317-323.

Wishart, J. "The Generalized Product Moment Distribution in Samples from a Normal Multivariate Population." Biometrika, 20 (1928), pp. 32-52.

Wright, S. "On the Nature of Size Factors." Genetics, 3 (1918), pp. 367-374.

Appendix A

First Derivatives of the Log Likelihood Function With Respect to the Parameter Matrices

Let: $D_{xx} = \frac{\partial \log l}{\partial \Sigma_{xx}}$ $\quad D_{xy} = \frac{\partial \log l}{\partial \Sigma_{xy}}$ $\quad D_{yx} = \frac{\partial \log l}{\partial \Sigma_{yx}}$ $\quad D_{yy} = \frac{\partial \log l}{\partial \Sigma_{yy}}$

$$\frac{\partial \log l}{\partial \Gamma} = 2(D_{xx}\Gamma + D_{xy}\Lambda(I-A)^{-1}B)\Sigma_{\zeta}$$

$$\frac{\partial \log l}{\partial \Sigma_{\zeta}} = 2[\Gamma' D_{xx}\Gamma + \Gamma' D_{xy}\Lambda(I-A)^{-1}B + B'(I-A)^{-t}\Lambda' D_{yx}\Gamma + B'(I-A)^{-t}\Lambda' D_{yy}\Lambda(I-A)^{-1}B]$$

$$-\mathrm{diag}[\Gamma' D_{xx}\Gamma + \Gamma' D_{xy}\Lambda(I-A)^{-1}B + B'(I-A)^{-t}\Lambda' D_{yx}\Gamma + B'(I-A)^{-t}\Lambda' D_{yy}\Lambda(I-A)^{-1}B]$$

$$\frac{\partial \log l}{\partial \Psi_{\omega}} = 2D_{xx} - \mathrm{diag}[D_{xx}]$$

$$\frac{\partial \log l}{\partial \Psi_{\varepsilon}} = 2D_{yy} - \mathrm{diag}[D_{yy}]$$

$$\frac{\partial \log l}{\partial \Sigma_{\theta}} = 2(I-A)^{-t}\Lambda' D_{yy}\Lambda(I-A)^{-1} - \mathrm{diag}[(I-A)^{-t}\Lambda' D_{yy}\Lambda(I-A)^{-1}]$$

$$\frac{\partial \log l}{\partial \Lambda} = 2[D_{yx}\Gamma\Sigma_{\zeta}\ B' + D_{yy}\Lambda(I-A)^{-1}(B\Sigma_{\zeta}B' + \Sigma_{\theta})](I-A)^{-t}$$

$$\frac{\partial \log l}{\partial B} = 2(I-A)^{-t}\Lambda'(D_{yx}\Gamma + D_{yy}\Lambda(I-A)^{-1}B)\Sigma_{\zeta}$$

$$\frac{\partial \log l}{\partial A} = 2(I-A)^{-t}\Lambda' D_{yy}\Lambda(I-A)^{-1}(B\Sigma_{\zeta}B' + \Sigma_{\theta})(I-A)^{-1}$$

$$+ (I-A)^{-t}\Lambda' D_{yx}\Gamma\Sigma_{\zeta}B'(I-A)^{-t}$$

$$+ (I-A)^{-1}B\Sigma_{\zeta}\Gamma' D_{xy}\Lambda(I-A)^{-1}$$

Let: $C_{xx} = \frac{\partial \log l}{\partial \Sigma_{xx_b}}$ $\quad C_{xy} = \frac{\partial \log l}{\partial \Sigma_{xy_b}}$ $\quad C_{yx} = \frac{\partial \log l}{\partial \Sigma_{yx_b}}$ $\quad C_{yy} = \frac{\partial \log l}{\partial \Sigma_{yy_b}}$

$$\frac{\partial \log l}{\partial \Gamma_b} = 2(C_{xx}\Gamma_b + C_{xy}\Lambda_b(I-A_b)^{-1}B_b)\Sigma_{\zeta_b}$$

$$\frac{\partial \log l}{\partial \Sigma_{\zeta_b}} = 2[\Gamma'_b C_{xx}\Gamma_b + \Gamma_b C_{xy}\Lambda_b(I-A_b)^{-1}B_b + B'_b(I-A_b)^{-t}\Lambda'_b C_{yx} + B'_b(I-A_b)^{-t}\Lambda'_b C_{yy}\Gamma_b$$

$$(I-A_b)^{-1}B_b] - \mathrm{diag}[\Gamma'_b C_{xx}\Gamma_b + \Gamma_b C_{xy}\Lambda_b(I-A_b)^{-1}B_b + B'_b(I-A_b)^{-t}\Lambda'_b C_{yx} +$$

$$B'_b(I-A_b)^{-t}\Lambda'_b C_{yy}\Lambda_b(I-A_b)^{-1}B_b]$$

$$\frac{\partial \log l}{\partial \Psi_{\omega_b}} = 2C_{xx} - \mathrm{diag}[C_{xx}]$$

$$\frac{\partial \log l}{\partial \Psi_{\varepsilon_b}} = 2C_{yy} - \mathrm{diag}[C_{yy}]$$

$$\frac{\partial \log l}{\partial \Sigma_{\theta_b}} = 2(I-A_b)^{-t}\Lambda'_b C_{yy}\Lambda_b(I-A_b)^{-1} - \mathrm{diag}[(I-A_b)^{-t}\Lambda'_b C_{yy}\Lambda_b(I-A_b)^{-1}]$$

$$\frac{\partial \log l}{\partial \Lambda_b} = 2[C_{yx}\Gamma_b\Sigma_{\zeta_b}B'_b + C_{yy}\Lambda_b(I-A_b)^{-1}(B_b\Sigma_{\zeta_b}B'_b + \Sigma_{\theta_b})][I-A_b]^{-t}$$

$$\frac{\partial \log l}{\partial B_b} = 2(I-A_b)^{-t}\Lambda'_b(C_{yx}\Gamma_b + C_{yy}\Lambda_b(I-A_b)^{-1}B_b)\Sigma_{\zeta_b}$$

$$\frac{\partial \log l}{\partial A_b} = 2(I-A_b)^{-t}\Lambda'_b C_{yy}\Lambda_b(I-A_b)^{-1}(B_b\Sigma_{\zeta_b}B'_b + \Sigma_{\theta_b})(I-A_b)^{-1}$$

$$+ (I-A_b)^{-t}\Lambda'_b C_{yx}\Gamma_b\Sigma_{\zeta_b}B'_b(I-A_b)^{-t}$$

$$+ (I-A_b)^{-1}B_b\Sigma_{\zeta}\Gamma'_b C_{xy}\Lambda_b(I-A_b)^{-1}$$

Appendix B

First Derivatives of Σ with Respect to Individual Elements of the Parameter Matrices

$$\frac{\partial \Sigma}{\partial \Gamma_{ij}} = \left[\begin{array}{c|c} \Gamma\Sigma_{\zeta}1_{ji} + 1_{ij}\Sigma_{\zeta}\Gamma' & 1_{ij}\Sigma_{\zeta}B'(I-A)^{-t}\Lambda' \\ \hline \Lambda(I-A)^{-1}B\Sigma_{\zeta}1_{ji} & 0 \end{array}\right]$$

$$\frac{\partial \Sigma}{\partial \Sigma_{\zeta_{ij}}} = \left[\begin{array}{c|c} \Gamma 1_{ij}^{ji}\Gamma' & \Gamma 1_{ij}^{ji}B'(I-A)^{-t}\Lambda' \\ \hline \Lambda(I-A)^{-1}B1_{ij}^{ji}\Gamma' & \Lambda(I-A)^{-1}B1_{ij}^{ji}B'(I-A)^{-t}\Lambda' \end{array}\right]$$

$$\frac{\partial \Sigma}{\partial \Psi_{\omega_{ij}}} = \left[\begin{array}{c|c} 1_{ij} & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial \Sigma}{\partial \Psi_{\varepsilon_{ij}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 1_{ij} \end{array}\right]$$

$$\frac{\partial \Sigma}{\partial \Sigma_{\theta_{ij}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & \Lambda(I-A)^{-1} 1_{ij}^{ji} (I-A)^{-t} \Lambda' \end{array}\right]$$

$$\frac{\partial \Sigma}{\partial \Lambda_{ij}} = \left[\begin{array}{c|c} 0 & \Gamma \Sigma_{\zeta} B' (I-A)^{-t} 1_{ji} \\ \hline 1_{ij} (I-A)^{-1} B \Sigma_{\zeta} \Gamma' & \left[\begin{array}{c} 1_{ij} (I-A)^{-1} (B \Sigma_{\zeta} B' + \Sigma_{\theta}) (I-A)^{-t} \Lambda' \\ + \\ \Lambda (I-A)^{-1} (B \Sigma_{\zeta} B' + \Sigma_{\theta}) (I-A)^{-t} 1_{ji} \end{array}\right] \end{array}\right]$$

$$\frac{\partial \Sigma}{\partial B_{ij}} = \left[\begin{array}{c|c} 0 & \Gamma \Sigma_{\zeta} B' 1_{ji} (I-A)^{-t} \Lambda' \\ \hline \Lambda (I-A)^{-1} 1_{ij} \Sigma_{\zeta} \Gamma' & \Lambda (I-A)^{-1} [1_{ij} \Sigma_{\zeta} B' + B \Sigma_{\zeta} 1_{ji}] (I-A)^{-t} \Lambda' \end{array}\right]$$

$$\frac{\partial\Sigma}{\partial A_{ij}} = \left[\begin{array}{c|c} 0 & \Gamma\Sigma_\zeta B'(I-A)^{-t}1_{ji}(I-A)^{-t}\Lambda' \\ \hline \Lambda(I-A)^{-1}1_{ij}(I-A)^{-1}B\Sigma_\zeta\Gamma' & \left[\begin{array}{c} \Lambda(I-A)^{-1}[(B\Sigma_\zeta B' + \Sigma_\theta)(I-A)^{-t}1_{ji} \\ + \\ 1_{ij}(I-A)^{-1}(B\Sigma_\zeta B' + \Sigma_\theta)](I-A)^{-t}\Lambda' \end{array}\right] \end{array}\right]$$

Appendix C

Second Derivatives of Σ with Respect to Individual Elements of the Parameter Matrices

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial \Gamma_{kl}} = \left[\begin{array}{c|c} 1_{kl}\Sigma_\zeta 1_{ji} + 1_{ij}\Sigma_\zeta 1_{lk} & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial \Sigma_{\zeta_{kl}}} = \left[\begin{array}{c|c} \Gamma 1_{kl}^{lk} 1_{ji} + 1_{ij} 1_{kl}^{lk} \Gamma' & 1_{ij} 1_{kl}^{lk} B'(1-A)^{-t} \Lambda' \\ \hline \Lambda(1-A)^{-1} B 1_{kl}^{lk} 1_{ji} & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial \Psi_{\omega_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial \Psi_{\varepsilon_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial \Sigma_{\theta_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial \Lambda_{kl}} = \left[\begin{array}{c|c} 0 & 1_{ij}\Sigma_\zeta B'(1-A)^{-t} 1_{lk} \\ \hline 1_{kl}(1-A)^{-1} B\Sigma_\zeta 1_{ji} & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial B_{kl}} = \left[\begin{array}{c|c} 0 & 1_{ij}\Sigma_\zeta 1_{lk}(1-A)^{-t}\Lambda' \\ \hline \Lambda(1-A)^{-1} 1_{kl}\Sigma_\zeta 1_{ji} & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Gamma_{ij} \partial A_{kl}} = \left[\begin{array}{c|c} 0 & 1_{ij}\Sigma_\zeta B'(1-A)^{-t} 1_{lk}(1-A)^{-t}\Lambda' \\ \hline \Lambda(1-A)^{-1} 1_{kl}(1-A)^{-1} B\Sigma_\zeta 1_{ji} & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\zeta_{ij}} \partial \Sigma_{\zeta_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\zeta_{ij}} \partial \Psi_{\omega_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\zeta_{ij}} \partial \Psi_{\varepsilon_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\zeta_{ij}} \partial \Sigma_{\theta_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\zeta_{ij}} \partial \Lambda_{kl}} = \left[\begin{array}{c|c} 0 & \Gamma 1_{ij}^{ji} B'(1-A)^{-t} 1_{lk} \\ \hline 1_{kl}(1-A)^{-1} B 1_{ij}^{ji} \Gamma' & \left[\begin{array}{c} 1_{kl}(1-A)^{-1} B 1_{ij}^{ji} B'(1-A)^{-t} \Lambda' \\ + \\ \Lambda(1-A)^{-1} B 1_{ij}^{ji} B'(1-A)^{-t} 1_{lk} \end{array}\right] \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\zeta_{ij}} \partial B_{kl}} = \left[\begin{array}{c|c} 0 & \Gamma 1^{ji}_{ij} 1_{lk} (1-A)^{-t} \Lambda' \\ \hline \Lambda (1-A)^{-1} 1_{kl} 1^{ji}_{ij} \Gamma' & \Lambda (1-A)^{-1} (1_{kl} 1^{ji}_{ij} B' + B 1^{ji}_{ij} 1_{lk}) (1-A)^{-t} \Lambda' \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\zeta_{ij}} \partial A_{kl}} = \left[\begin{array}{c|c} 0 & \Gamma 1^{ji}_{ij} B' (1-A)^{-t} 1_{lk} (1-A)^{-t} \Lambda' \\ \hline \Lambda (1-A)^{-1} 1_{kl} (1-A)^{-1} B 1^{ji}_{ij} \Gamma' & \left[\begin{array}{c} \Lambda (1-A)^{-1} (1_{kl} (1-A)^{-1} B 1^{ij}_{ji} B' \\ + \\ B 1^{ji}_{ij} B' (1-A)^{-t} 1_{kl}) (1-A)^{-t} \Lambda' \end{array}\right] \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\omega_{ij}} \partial \Psi_{\omega_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\omega_{ij}} \partial \Psi_{\varepsilon_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\omega_{ij}} \partial \Sigma_{\theta_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\omega_{ij}} \partial \Lambda_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\omega_{ij}} \partial B_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\omega_{ij}} \partial A_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\varepsilon_{ij}} \partial \Psi_{\varepsilon_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\varepsilon_{ij}} \partial \Sigma_{\theta_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\varepsilon_{ij}} \partial \Lambda_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\varepsilon_{ij}} \partial B_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Psi_{\varepsilon_{ij}} \partial A_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\theta_{ij}} \partial \Sigma_{\theta_{kl}}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\theta_{ij}} \partial \Lambda_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & \left[\begin{array}{c} 1_{kl}(1-A)^{-1} 1_{ij}^{ji} (1-A)^{-t} \Lambda' \\ + \\ \Lambda (1-A)^{-1} 1_{ij}^{ji} (1-A)^{-t} 1_{lk} \end{array}\right] \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\theta_{ij}} \partial B_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Sigma_{\theta_{ij}} \partial A_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & \left[\begin{array}{c} \Lambda (1-A)^{-1} (1_{kl} (1-A)^{-1} 1_{ij}^{ji} \\ + \\ 1_{ij}^{ji} (1-A)^{-t} 1_{lk}) (1-A)^{-t} \Lambda' \end{array}\right] \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Lambda_{ij} \partial \Lambda_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & \left[\begin{array}{c} 1_{ij}(1-A)^{-1}(B\Sigma_\zeta B' + \Sigma_\theta)(1-A)^{-t} 1_{lk} \\ + \\ 1_{kl}(1-A)^{-1}(B\Sigma_\zeta B' + \Sigma_\theta)(1-A)^{-t} 1_{ji} \end{array}\right] \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Lambda_{ij} \partial B_{kl}} = \left[\begin{array}{c|c} 0 & \Gamma\Sigma_\zeta 1_{lk}(1-A)^{-t} 1_{ij} \\ \hline 1_{ij}(1-A)^{-1} 1_{kl}\Sigma_\zeta\Gamma' & \left[\begin{array}{c} 1_{ij}(1-A)^{-1}[1_{kl}\Sigma_\zeta B' + B\Sigma_\zeta 1_{lk}](1-A)^{-t}\Lambda' \\ + \\ \Lambda(1-A)^{-1}[1_{kl}\Sigma_\zeta B' + B\Sigma_\zeta 1_{lk}](1-A)^{-t} 1_{ji} \end{array}\right] \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial \Lambda_{ij} \partial A_{kl}} = \left[\begin{array}{c|c} 0 & \Gamma \Sigma_\zeta B'(1-A)^{-t} 1_{lk} (1-A)^{-t} 1_{ij} \\ \hline 1_{ij}(1-A)^{-1} 1_{kl}(1-A)^{-1} B \Sigma_\zeta \Gamma' & \left[\begin{array}{c} 1_{ij}(1-A)^{-1}(1_{kl}(1-A)^{-1}(B\Sigma_\zeta B'+\Sigma_\theta) \\ + (B\Sigma_\zeta B'+\Sigma_\theta)(1-A)^{-t} 1_{lk})(1-A)^{-t}\Lambda' \\ + \\ \Lambda(1-A)^{-1}(1_{kl}(1-A)^{-1}(B\Sigma_\zeta B'+\Sigma_\theta) \\ + (B\Sigma_\zeta B'+\Sigma_\theta)(1-A)^{-t} 1_{lk})(1-A)^{-t} 1_{ji} \end{array}\right] \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial B_{ij} \partial B_{kl}} = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & \Lambda(1-A)^{-1}[1_{ij}\Sigma_\zeta 1_{lk} + 1_{kl}\Sigma_\zeta 1_{ji}](1-A)^{-t}\Lambda' \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial B_{ij}\partial A_{kl}} = \left[\begin{array}{c|c} 0 & \Gamma\Sigma_\zeta 1_{ji}(I-A)^{-t}1_{lk}(I-A)^{-1}\Lambda' \\ \hline \Lambda(I-A)^{-1}1_{kl}(I-A)^{-1}1_{ij}\Sigma_\zeta\Gamma' & \Lambda(I-A)^{-1}\left[\begin{array}{c} (1_{ij}\Sigma_\zeta B' + B\Sigma_\zeta 1_{ji})(I-A)^{-t}1_{lk} \\ + \\ 1_{kl}(I-A)^{-1}(1_{ij}\Sigma_\zeta B' + B\Sigma_\zeta 1_{ji}) \end{array}\right](I-A)^{-t}\Lambda' \end{array}\right]$$

$$\frac{\partial^2 \Sigma}{\partial A_{ij}\partial A_{kl}} = \left[\begin{array}{c:c} 0 & \begin{array}{l} \Gamma\Sigma_\zeta B'(1-A)^{-t}(1_{lk}(1-A)^{-t}1_{ji} + \\ 1_{ji}(1-A)^{-t}1_{lk})(1-A)^{-t}\Lambda' \end{array} \\ \hdashline \begin{array}{l} \Lambda(1-A)^{-1}(1_{ij}(1-A)^{-1}1_{kl}+ \\ 1_{kl}(1-A)^{-1}1_{ij})(1-A)^{-1}B\Sigma_\zeta\Gamma' \end{array} & \Lambda(1-A)^{-1}\left[\begin{array}{c} 1_{kl}(1-A)^{-1}[(1_{ij}(1-A)^{-1}(B\Sigma_\zeta B'+\Sigma_\theta) + (B\Sigma_\zeta B'+\Sigma_\theta)(I-A)^{-t}1_{ji}] \\ + \\ 1_{ij}(1-A)^{-1}(B\Sigma_\zeta B' + \Sigma_\theta) + (B\Sigma_\zeta B'+\Sigma_\theta)(I-A)^{-t}1_{ji}](I-A)^{-t}1_{lk} \\ + \\ 1_{ij}(1-A)^{-1}1_{kl}(1-A)^{-1}(B\Sigma_\zeta B'+\Sigma_\theta) \\ + \\ (B\Sigma_\zeta B'+\Sigma_\theta)(1-A)^{-t}1_{lk}(I-A)^{-t}1_{ji} \end{array}\right](1-A)^{-t}\Lambda' \end{array}\right]$$

Appendix D

Deck Setup for Use of Standard Error Routine

```
ATTACH,X,JOESSEPROGRAMLGO.
REWIND,X.
X.
*EOR
> Title Card < (required)
> Main Parameter Card < (required)
> Parameter Set < (1 required for each non-zero parameter matrix)
99                 (required to terminate reading of parameter sets)
*EOR
```

<u>Title Card</u>

Descriptive information printed at the start of each job

<u>Main Parameter Card</u>

Consists of eight fields of three digits each, right justify all information

1. number of groups
2. number of subjects per group
3. number of observed exogenous variables
4. number of observed endogenous variables
5. number of latent between groups exogenous variables
6. number of latent within groups exogenous variables
7. number of latent between groups endogenous variables
8. number of latent within groups endogenous variables

Parameter Sets

One set of these is required for each non-zero or non-fixed parameter matrix.

Each set is composed of five items:

1. A parameter identification card containing a number from 1 to 16 right justified in columns 1-2.
2. A format card describing a row of the parameter matrix.
3. One card for each row of the parameter matrix containing the estimated (or fixed) values for the elements in that row.
4. A format card describing a row of the parameter specification matrix.
5. One card for each row of the parameter specification matrix.

The correspondence between the parameter ID numbers and the specific parameter matrix is set forth in Table 1.

The format card describing the rows of the parameter matrix should contain only F or E formats.

The format card describing the rows of the parameter specification matrix should contain only I formats.

The parameter matrix should be presented as a rectangular matrix (e.g., symmetric matrices cannot appear as lower triangular).

The parameter specification matrix is obtainable from the printout of the estimation routine and should consist of only 0's (for fixed elements) and integers ranging from 1 to the total number of unique parameter estimates. Elements which have been constrained to be equal should have the same number.

Table 1

Correspondence Between Parameter ID Numbers and Parameter Matrices

Number	Matrix
1	A
2	B
3	Λ
4	Σ_θ
5	Σ_ζ
6	Ψ_ε
7	Ψ_ω
8	Γ
9	A_b
10	B_b
11	Λ_b
12	Σ_{θ_b}
13	Σ_{ζ_b}
14	Ψ_{ε_b}
15	Ψ_{ω_b}
16	Γ_b

Appendix E

Listing of Standard Error Program

```
 PROGRAM SE(INPUT,OUTPUT,TAPE1=INPUT,TAPE2=OUTPUT)
 INTEGER OX,OY
 INTEGER CA,CB,CL,CST,CSX,CPE,CPW,CG,CAB,CBB,CLB,CSTB,
+CSXB,CPEB,CPWB,CGB
 REAL L,LB,KMAT
 COMMON /BLK1/A(200,1),B(200,1),L(200,1),ST(200,1),
+PE(200,1),PW(200,1),SX(200,1),G(200,1),
+CA(200,1),CB(200,1),CL(200,1),CST(200,1),
+CPE(200,1),CPW(200,1),CSX(200,1),CG(200,1)
 COMMON /BLK2/AB(200,1),BB(200,1),LB(200,1),STB(200,1),
+PEB(200,1),PWB(200,1),SXB(200,1),GB(200,1),
+CAB(200,1),CBB(200,1),CLB(200,1),CSTB(200,1),
+CPEB(200,1),CPWB(200,1),CSXB(200,1),CGB(200,1)
 COMMON /BLK3/ LX,LY
 COMMON /BLK4/ LXB,LYB
 COMMON /BLK5/ IMATRIX(80),ICOL(80),IROW(80),IP(80),
+JROW(16),JCOL(16),OX,OY,M,N
 COMMON /BLK6/ SBXX(120,1),SBYY(120,1),SBYX(225,1)
 COMMON /BLK7/ SWXX(120,1),SWYY(120,1),SWYX(225,1)
 COMMON /BLK8/ SWIXX(120,1),SWIYY(120,1),SWIYX(225,1)
 COMMON /BLK9/ SBWIXX(120,1),SBWIYY(120,1),SBWIYX(225,1)
 COMMON /BLK10/ TMP1(1500,1),TMP2(1500,1),TMP3(500,1),
+TMP4(500,1),TMP5(500,1),TMP6(500,1)
 COMMON /BLK11/ FD1XX(120,1),FD1YY(120,1),FD1YX(225,1)
 COMMON /BLK12/ FD2XX(120,1),FD2YY(120,1),FD2YX(225,1)
 DIMENSION X(3200,1),Y(3200,1),IJK(356)
 DIMENSION KMAT(1500,1),ESND(829)
 DIMENSION HEAD(20)
 EQUIVALENCE (X,A),(Y,AB),(IJK,IMATRIX)
 EQUIVALENCE (KMAT,TEMP4)
 DATA X/3200*0.0/,Y/3200*0.0/,IJK/356*0/
```

```
      DATA SBXX,SBYY,SBYX/465*0.0/
      DATA SWXX,SWYY,SWYX/465*0.0/
      READ(1,100) TITLE
100   FORMAT(8A10)
      WRITE(2,101) TITLE
101   FORMAT(1X,8A10)
      READ(1,102) M,N,OX,OY,LXB,LX,LYB,LY
102   FORMAT(8I3)
      WRITE(2,103) M,N,OX,OY,LXB,LX,LYB,LY
103   FORMAT(8(1X,I3))
      JROW(1)=LY
      JROW(2)=LY
      JROW(3)=OY
      JROW(4)=LY
      JROW(5)=LX
      JROW(6)=OY
      JROW(7)=OX
      JROW(8)=OX
      JROW(9)=LYB
      JROW(10)=LYB
      JROW(11)=OY
      JROW(12)=LYB
      JROW(13)=LXB
      JROW(14)=OY
      JROW(15)=OX
      JROW(16)=OX
      JCOL(1)=LY
      JCOL(2)=LX
      JCOL(3)=LY
      JCOL(4)=LY
      JCOL(5)=LX
```

```
      JCOL(6)=OY
      JCOL(7)=OX
      JCOL(8)=LX
      JCOL(9)=LYB
      JCOL(10)=LXB
      JCOL(11)=LYB
      JCOL(12)=LYB
      JCOL(13)=LXB
      JCOL(14)=OY
      JCOL(15)=OX
      JCOL(16)=LXB
      WRITE(2,1155) (JROW(I),JCOL(I),I=1,16)
1155  FORMAT(1X,I4,1X,I4)
1     READ(1,2) IPARM
2     FORMAT(I2)
      WRITE(2,1155) IPARM
      IF(IPARM.EQ.99) GO TO 3
      IF(IPARM.LE.8) CALL RD1(IPARM)
      IF(IPARM.GT.8) CALL RD2(IPARM)
      GO TO 1
C DEFINE IMATRIX,IROW,ICOL,IP
3     LL=0
      CALL MRCP(CA,LY,LY,LL,1)
      CALL MRCP(CB,LY,LX,LL,2)
      CALL MRCP(CL,OY,LY,LL,3)
      CALL MRCP(CST,LY,LY,LL,4)
      CALL MRCP(CSX,LX,LX,LL,5)
      CALL MRCP(CPE,OY,OY,LL,6)
      CALL MRCP(CPW,OX,OX,LL,7)
      CALL MRCP(CG,OX,LX,LL,8)
      CALL MRCP(CAB,LYB,LYB,LL,9)
```

```
      CALL MRCP(CBB,LYB,LXB,LL,10)
      CALL MRCP(CLB,OY,LYB,LL,11)
      CALL MRCP(CSTB,LYB,LYB,LL,12)
      CALL MRCP(CSXB,LXB,LXB,LL,13)
      CALL MRCP(CPEB,OY,OY,LL,14)
      CALL MRCP(CPWB,OX,OX,LL,15)
      CALL MRCP(CGB,OX,LXB,LL,16)
      WRITE(2,1156) (IMATRIX(I),ICOL(I),IROW(I),IP(I),I=1,20)
1156  FORMAT(4(1X,I3))
      NPAR=LL
      NUPAR=IP(LL)
      WRITE(2,1155) NPAR,NUPAR
      CALL HAT(AB,BB,GB,LB,STB,SXB,PEB,PWB,SBXX,SBYX,SBYY,OX,OY,
     +LXB,LYB)
      CALL HAT(A,B,G,L,ST,SX,PE,PW,SWXX,SWYX,SWYY,OX,OY,LX,LY)
      IF(OX.EQ.0)
     +CALL ISMSL(OY,SWYY,SWIYY,TMP1,DET,5.E-12,IERR)
      WRITE(2,99199)(SWXX(I),SWYX(I),SWYY(I),I=1,10)
99199 FORMAT(3(1X,F10.5))
      IF(OX.NE.0)
     +CALL IPSMSL(OX,OY,SWXX,SWYX,SWYY,5.E-12,SWIXX,SWIYX,SWIYY,
     +DET,IERR,TMP1,TMP2,TMP3)
      ANUM=N
      CALL SCM(SBYY,ANUM,TMP2,OY,OY,1)
      CALL ADD(SWYY,TMP2,TMP1,OY,OY,1)
      IF(OX.EQ.0) GO TO 300
      CALL SCM(SBXX,ANUM,TMP3,OX,OX,1)
      CALL ADD(SWXX,TMP3,TMP2,OX,OX,1)
      CALL SCM(SBYX,ANUM,TMP4,OY,OX,0)
      CALL ADD(SWYX,TMP4,TMP3,OY,OX,0)
      WRITE(2,99199) (TMP2(I),TMP3(I),TMP1(I),I=1,10)
```

```
      CALL IPSMSL(OX,OY,TMP2,TMP3,TMP1,5.E-12,SBWIXX,SBWIYX,SBWIYY,
     +DET,IERR,TMP4,TMP5,TMP6)
      GO TO 301
300   CALL ISMSL(OY,TMP1,SBWIYY,TMP2,DET,5.E-12,IERR)
301   CONTINUE
      K=1
      DO 109 II=1,NPAR
C      WRITE(2,9988)
9988  FORMAT(1X,1HZ)
      DO 109 JJ=1,II
      TR1=0.0
      TR2=0.0
      TRSND=0.0
      IF(IMATRIX(II).LE.8) CALL SND1(II,JJ,TRSND,LX,LY,A,B,L,ST,PE,PW,SX
     +,G)
      IF(IMATRIX(JJ).GT.8) CALL SND1(II,JJ,TRSND,LXB,LYB,AB,BB,LB,STB,PE
     +B,PWB,SXB,GB)
      IF(IMATRIX(JJ).GT.8) TRSND=TRSND*N
      IF(IMATRIX(II).GT.8) GO TO 201
      CALL FST1(II,FD1XX,FD1YY,FD1YX)
      GO TO 202
201   CALL FST2(II,FD1XX,FD1YY,FD1YX)
202   IF(IMATRIX(JJ).GT.8) GO TO 203
      CALL FST1(JJ,FD2XX,FD2YY,FD2YX)
      GO TO 204
203   CALL FST2(JJ,FD2XX,FD2YY,FD2YX)
204   CALL ITRACE(SBWIXX,SBWIYX,SBWIYY,FD1XX,FD1YX,FD1YY,
     +FD2XX,FD2YX,FD2YY,TR1)
      IF(IMATRIX(II).GT.8) GO TO 205
      CALL ITRACE(SWIXX,SWIYX,SWIYY,FD1XX,FD1YX,FD1YY,FD2XX,
     +FD2YX,FD2YY,TR2)
205   ESND(K)=.5*TRSND+(M-2)*.5*TR1-(M-M*N)*.5*TR2
109   K=K+1
```

```
      WRITE(2,1167)
1167  FORMAT(1X,1HH)
C COMPUTE K MATRIX
      DO 400 I=1,NPAR
      DO 400 J=1,NUPAR
      ILOC=(J-1)*NPAR+I
      KMAT(ILOC)=0
      IF(IP(I).EQ.J) KMAT(ILOC)=1
400   CONTINUE
      CALL MPYTR(KMAT,ESND,TMP1,NPAR,NUPAR,0,1,NPAR)
      CALL MPY(TMP1,KMAT,TMP2,NUPAR,NPAR,0,0,NUPAR)
      CALL INVS(TMP2,NUPAR,DET,TMP1,TMP3)
      XLB=10H
      NPAGE=1
      HEAD(1)=10H
      CALL PRINT(TMP2,NUPAR,NUPAR,0,XLB,XLB,NPAGE,HEAD,
     +40HVARIANCE-COVARIANCE MATRIX OF ESTIMATES )
      WRITE(2,1168)
1168  FORMAT(1X,1HI)
      END
      SUBROUTINE MRCP(JMAT,IR,IC,L,K)
      COMMON /BLK5/ IMATRIX(80),ICOL(80),IROW(80),IP(80),
     +JROW(16),JCOL(16),OX,OY,M,N
      DIMENSION JMAT(1,1)
      IF(IR.EQ.0.OR.IC.EQ.0) GO TO 2
      DO 1 I=1,IR
      DO 1 J=1,IC
      INUM=(J-1)*IR+I
      IF(JMAT(INUM).EQ.0) GO TO 1
      L=L+1
      IMATRIX(L)=K
```

```
      IROW(L)=I
      ICOL(L)=J
      IP(L)=JMAT(INUM)
1     CONTINUE
2     RETURN
      END
      SUBROUTINE ITRACE(SBWIXX,SBWIYX,SBWIYY,FD1XX,FD1YX,
     +FD1YY,FD2XX,FD2YX,FD2YY,TR)
      INTEGER OX,OY
      COMMON /BLK5/ IMATRIX(80),ICOL(80),IROW(80),IP(80),
     +JROW(16),JCOL(16),OX,OY,M,N
      COMMON /BLK10/ TMP1(1500,1),TMP2(1500,1),TMP3(500,1),
     +TMP4(500,1),TMP5(500,1),TMP6(500,1)
      DIMENSION SBWIXX(1,1),SBWIYY(1,1),SBWIYX(1,1),
     +FD1XX(1,1),FD1YY(1,1),FD1YX(1,1),FD2XX(1,1),
     +FD2YY(1,1),FD2YX(1,1)
      IF(OX.EQ.0) GO TO 1
      CALL MPY(SBWIXX,FD1XX,TMP1,OX,OX,1,0,OX)
      CALL MPYTR(SBWIYX,FD1YX,TMP2,OY,OX,0,0,OX)
      CALL ADD(TMP1,TMP2,TMP3,OX,OX,0)
      CALL MPY(SBWIXX,FD2XX,TMP1,OX,OX,1,0,OX)
      CALL MPYTR(SBWIYX,FD2YX,TMP2,OY,OX,0,0,OX)
      CALL ADD(TMP1,TMP2,TMP4,OX,OX,0)
      CALL MPY(TMP3,TMP4,TMP1,OX,OX,0,0,OX)
      CALL TRACE(TMP1,TR1,OX,0)
      CALL MPYRT(SBWIXX,FD1YX,TMP1,OX,OX,1,0,OY)
      CALL MPYTR(SBWIYX,FD1YY,TMP2,OY,OX,0,0,OY)
      CALL ADD(TMP1,TMP2,TMP3,OX,OY,0)
      CALL MPY(SBWIYX,FD2XX,TMP1,OY,OX,0,0,OX)
      CALL MPY(SBWIYY,FD2YX,TMP2,OY,OY,1,0,OX)
      CALL ADD(TMP1,TMP2,TMP4,OY,OX,0)
```

```
      CALL MPY(TMP3,TMP4,TMP1,OX,OY,0,0,OX)
      CALL TRACE(TMP1,TR2,OX,0)
      CALL MPY(SBWIYX,FD1XX,TMP1,OY,OX,0,0,OX)
      CALL MPY(SBWIYY,FD1YX,TMP2,OY,OY,1,0,OX)
      CALL ADD(TMP1,TMP2,TMP3,OY,OX,0)
      CALL MPYRT(SBWIXX,FD2YX,TMP1,OX,OX,1,0,OY)
      CALL MPYTR(SBWIYX,FD2YY,TMP2,OY,OX,0,0,OY)
      CALL ADD(TMP1,TMP2,TMP4,OX,OY,0)
      CALL MPY(TMP3,TMP4,TMP1,OY,OX,0,0,OY)
      CALL TRACE(TMP1,TR3,OY,0)
      CALL MPYRT(SBWIYX,FD1YX,TMP1,OY,OX,0,0,OY)
      CALL MPY(SBWIYY,FD1YY,TMP2,OY,OY,1,0,OY)
      CALL ADD(TMP1,TMP2,TMP3,OY,OY,0)
      CALL MPYRT(SBWIYX,FD2YX,TMP1,OY,OX,0,0,OY)
      CALL MPY(SBWIYY,FD2YY,TMP2,OY,OY,1,0,OY)
      CALL ADD(TMP1,TMP2,TMP4,OY,OY,0)
      CALL MPY(TMP3,TMP4,TMP1,OY,OY,0,0,OY)
      CALL TRACE(TMP1,TR4,OY,0)
      TR=TR1+TR2+TR3+TR4
      RETURN
1     CONTINUE
      CALL MPY(SBWIYY,FD1YY,TMP1,OY,OY,1,0,OY)
      CALL MPY(SBWIYY,FD2YY,TMP2,OY,OY,1,0,OY)
      CALL MPY(TMP1,TMP2,TMP3,OY,OY,0,0,OY)
      CALL TRACE(TMP3,TR,OY,0)
      RETURN
      END
      SUBROUTINE HAT(A,B,G,L,ST,SX,PE,PW,SBXX,SBYX,SBYY,OX,OY,
     +LX,LY)
      DIMENSION A(1,1),B(1,1),L(1,1),ST(1,1),SX(1,1),PE(1,1),
     +PW(1,1),SBXX(1,1),SBYX(1,1),SBYY(1,1)
```

```
      COMMON /BLK10/ TMP1(1500,1),TMP2(1500,1),TMP3(500,1),
     +TMP4(500,1),TMP5(500,1),TMP6(500,1)
      IF(OX.EQ.0) GO TO 1
      CALL MPY(G,SX,TMP1,OX,LX,0,0,LX)
      CALL MPYRT(TMP1,G,TMP2,OX,LX,0,0,OX)
      CALL ADD(TMP2,PW,TMP1,OX,OX,0)
      CALL ALTER(TMP1,SBXX,OX)
1     CONTINUE
      CALL SCMA(TMP1,1.0,LY,0)
      CALL SUB(TMP1,A,TMP2,LY,LY,0)
      CALL INVS(TMP2,LY,DET,TMP5,TMP6)
      IF(OX.EQ.0) GO TO 2
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,SBYX,OY,LX,0,0,OX)
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL ADD(TMP3,ST,TMP1,LY,LY,0)
2     IF(OX.EQ.0) CALL SCM(ST,1.0,TMP1,LY,LY,0)
      CALL MPY(L,TMP2,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP1,TMP4,OY,LY,0,0,LY)
      CALL MPYRT(TMP4,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,TMP2,OY,LY,0,0,OY)
      CALL ADD(TMP2,PE,TMP1,OY,OY,0)
      CALL ALTER(TMP1,SBYY,OY)
      RETURN
      END
      SUBROUTINE ALTER(M1,M2,ISZ)
      DIMENSION M1(1) , M2(1)
      L=1
```

```
      CALL READ(CPE,OY,OY,0,1)
      RETURN
7     CALL READ(PW,OX,OX,0,1)
      CALL READ(CPW,OX,OX,0,1)
      RETURN
8     CALL READ(G,OX,LX,0,1)
      CALL READ(CG,OX,LX,0,1)
      RETURN
      END
      SUBROUTINE RD2(IPARM)
      COMMON /BLK2/A(200,1),B(200,1),L(200,1),ST(200,1),
     +PE(200,1),PW(200,1),SX(200,1),G(200,1),
     +CA(200,1),CB(200,1),CL(200,1),CST(200,1),
     +CPE(200,1),CPW(200,1),CSX(200,1),CG(200,1)
      COMMON /BLK4/ LX,LY
      COMMON /BLK5/ IMATRIX(80),ICOL(80),IROW(80),IP(80),
     +JROW(16),JCOL(16),OX,OY,M,N
      KPARM=IPARM-8
      GO TO (1,2,3,4,5,6,7,8) KPARM
1     CALL READ(A,LY,LY,0,1)
      CALL READ(CA,LY,LY,0,1)
      RETURN
2     CALL READ(B,LY,LX,0,1)
      CALL READ(CB,LY,LX,0,1)
      RETURN
3     CALL READ(L,OY,LY,0,1)
      CALL READ(CL,OY,LY,0,1)
      RETURN
4     CALL READ(ST,LY,LY,0,1)
      CALL READ(CST,LY,LY,0,1)
      RETURN
5     CALL READ(SX,LX,LX,0,1)
```

```
      CALL READ(CSX,LX,LX,0,1)
      RETURN
6     CALL READ(PE,OY,OY,0,1)
      CALL READ(CPE,OY,OY,0,1)
      RETURN
7     CALL READ(PW,OX,OX,0,1)
      CALL READ(CPW,OX,OX,0,1)
      RETURN
8     CALL READ(G,OX,LX,0,1)
      CALL READ(CG,OX,LX,0,1)
      RETURN
      END
      SUBROUTINE SND1(IN1,IN2,TRSND,LX,LY,A,B,L,ST,PE,PW,SX,G)
      INTEGER OX,OY
      REAL L,LB
      COMMON /BLK5/ IMATRIX(80),ICOL(80),IROW(80),IP(80),
     +JROW(16),JCOL(16),OX,OY,M,N
      COMMON /BLK9/ SBWIXX(120,1),SBWIYY(120,1),SBWIYX(225,1)
      COMMON /BLK10/ TMP1(1500,1),TMP2(1500,1),TMP3(500,1),
     +TMP4(500,1),TMP5(500,1),TMP6(500,1)
      COMMON /BLK11/ SDXX(120,1),SDYY(120,1),SDYX(225,1)
      DIMENSION ONEIJ(20,20),ONEKL(20,20),ONEIJJI(20,20),ONE(20,20)
      DIMENSION A(1),B(1),L(1),ST(1),PE(1),PW(1),SX(1),G(1)
      IPARM=IN1
      JPARM=IN2
      ILSS=0
      IF(IMATRIX(IPARM).GT.8) ILSS=8
      TRSND=0.0
      IF(OX.EQ.0) GO TO 1
      CALL GEN(SDXX,0.0,OX,OX,0)
      CALL GEN(SDYX,0.0,OY,OX,0)
1     CONTINUE
```

```
      CALL GEN(SDYY,0.0,OY,OY,0)
      CALL SCMA(TMP1,1.0,LY,0)
      CALL SUB(TMP1,A,TMP2,LY,LY,0)
      CALL INVS(TMP2,LY,DET,TMP5,TMP6)
      CALL GEN(ONEIJ,0.0,JROW(IMATRIX(IPARM)),JCOL(IMATRIX(IPARM)),0)
      INUM=(ICOL(IPARM)-1)*JROW(IMATRIX(IPARM))+IROW(IPARM)
      ONEIJ(INUM)=1.0
      CALL GEN(ONEKL,0.0,JROW(IMATRIX(JPARM)),JCOL(IMATRIX(JPARM)),0)
      INUM=(ICOL(JPARM)-1)*JROW(IMATRIX(JPARM))+IROW(JPARM)
      ONEKL(INUM)=1.0
      LMAX=OX
      IF(OX.LT.OY) LMAX=OY
      CALL SCMA(ONE,1.0,LMAX,2)
      GO TO(101,102,103,104,107,105,106,108) IMATRIX(IPARM)-ILSS
101   IF(OX.EQ.0) GO TO 2
      CALL MPY(ONEIJ,TMP2,TMP1,LY,LY,0,0,LY)
      CALL MPY(TMP1,ONEKL,TMP3,LY,LY,0,0,LY)
      CALL MPY(ONEKL,TMP2,TMP4,LY,LY,0,0,LY)
      CALL MPY(TMP4,ONEIJ,TMP5,LY,LY,0,0,LY)
      CALL ADD(TMP3,TMP5,TMP1,LY,LY,0)
      CALL MPY(L,TMP2,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP1,TMP4,OY,LY,0,0,LY)
      CALL MPY(TMP4,TMP2,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,B,TMP1,OY,LY,0,0,LX)
      CALL MPY(TMP1,SX,TMP3,OY,LX,0,0,LX)
      CALL MPYRT(TMP3,G,SDYX,OY,LX,0,0,OX)
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL ADD(TMP3,ST,TMP1,LY,LY,0)
2     IF(OX.EQ.0) CALL SCM(ST,1.0,TMP1,LY,LY,0)
      CALL MPYRT(TMP1,TMP2,TMP3,LY,LY,0,0,LY)
```

```
      CALL MPYRT(TMP3,ONEIJ,TMP4,LY,LY,0,0,LY)
      CALL TRNSP(TMP4,TMP3,LY,LY)
      CALL ADD(TMP3,TMP4,TMP5,LY,LY,0)
      CALL MPY(ONEKL,TMP2,TMP3,LY,LY,0,0,LY)
      CALL MPY(TMP3,TMP5,TMP4,LY,LY,0,0,LY)
      CALL TRNSP(TMP4,TMP5,LY,LY)
      CALL ADD(TMP4,TMP5,TMP3,LY,LY,0)
      CALL MPY(TMP2,TMP1,TMP4,LY,LY,0,0,LY)
      CALL MPY(ONEKL,TMP4,TMP1,LY,LY,0,0,LY)
      CALL MPY(TMP2,TMP1,TMP5,LY,LY,0,0,LY)
      CALL MPY(ONEIJ,TMP5,TMP6,LY,LY,0,0,LY)
      CALL TRNSP(TMP6,TMP4,LY,LY)
      CALL ADD(TMP4,TMP6,TMP1,LY,LY,0)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,TMP4,TMP5,OY,LY,0,0,LY)
      CALL MPYRT(TMP5,TMP1,SDYY,OY,LY,0,0,OY)
1000  IF(OX.EQ.0) GO TO 1003
      CALL MPY(SDXX,SBWIXX,TMP6,OX,OX,0,1,OX)
      CALL MPYRT(SDYX,SBWIYX,TMP4,OY,OX,0,0,OY)
      CALL MPYTR(SDYX,SBWIYX,TMP3,OY,OX,0,0,OX)
      CALL MPY(SDYY,SBWIYY,TMP5,OY,OY,0,1,OY)
      CALL TRACE(TMP3,ATR,OX,0)
      CALL TRACE(TMP4,BTR,OY,0)
      CALL TRACE(TMP5,CTR,OY,0)
      CALL TRACE(TMP6,DTR,OX,0)
      TRSND=ATR+BTR+CTR+DTR
1010  RETURN
1003  CALL MPY(SDYY,SBWIYY,TMP5,OY,OY,0,1,OY)
      CALL TRACE(TMP5,TRSND,OY,0)
      RETURN
102   GO TO(201,202) IMATRIX(JPARM)-ILSS
201   CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
```

```
      CALL MPY(TMP1,ONEKL,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEIJ,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,SDYX,OY,LX,0,0,OX)
      CALL MPY(ONEIJ,SX,TMP3,LY,LX,0,0,LX)
      CALL MPYRT(TMP3,B,TMP4,LY,LX,0,0,LY)
      CALL TRNSP(TMP4,TMP5,LY,LY)
      CALL ADD(TMP4,TMP5,TMP3,LY,LY,0)
      CALL MPYRT(TMP3,TMP2,TMP1,LY,LY,0,0,LY)
      CALL MPYRT(TMP1,ONEKL,TMP3,LY,LY,0,0,LY)
      CALL TRNSP(TMP3,TMP4,LY,LY)
      CALL ADD(TMP3,TMP4,TMP5,LY,LY,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,TMP5,TMP4,OY,LY,0,0,LY)
      CALL MPYRT(TMP4,TMP1,SDYY,OY,LY,0,0,OY)
      GO TO 1000
202   CALL MPY(ONEIJ,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEKL,TMP3,LY,LX,0,0,LY)
      CALL TRNSP(TMP3,TMP4,LY,LY)
      CALL ADD(TMP3,TMP4,TMP1,LY,LY,0)
      CALL MPY(L,TMP2,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP1,TMP4,OY,LY,0,0,LY)
      CALL MPYRT(TMP4,TMP2,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,L,SDYY,OY,LY,0,0,OY)
      GO TO 1000
103   GO TO(203,204,205) IMATRIX(JPARM)-ILSS
203   IF(OX.EQ.0) GO TO 3
      CALL MPY(ONEIJ,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEKL,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
```

```
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,SDYX,OY,LX,0,0,OX)
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP4,LY,LX,0,0,LY)
      CALL ADD(TMP4,ST,TMP3,LY,LY,0)
3     IF(OX.EQ.0) CALL SCM(ST,1.0,TMP3,LY,LY,0)
      CALL MPY(ONEKL,TMP2,TMP4,LY,LY,0,0,LY)
      CALL MPY(TMP4,TMP3,TMP1,LY,LY,0,0,LY)
      CALL TRNSP(TMP1,TMP3,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(TMP2,TMP4,TMP5,LY,LY,0,0,LY)
      CALL MPY(ONEIJ,TMP5,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP5,OY,LY,0,0,LY)
      CALL MPYRT(TMP5,L,TMP3,OY,LY,0,0,OY)
      CALL TRNSP(TMP3,TMP4,OY,OY)
      CALL ADD(TMP3,TMP4,SDYY,OY,OY,0)
      GO TO 1000
204   CALL MPY(ONEIJ,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEKL,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,SDYX,OY,LX,0,0,OX)
      CALL MPY(ONEKL,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL TRNSP(TMP3,TMP1,LY,LY)
      CALL ADD(TMP3,TMP1,TMP4,LY,LY,0)
      CALL MPY(TMP2,TMP4,TMP1,LY,LY,0,0,LY)
      CALL MPY(ONEIJ,TMP1,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,TMP4,OY,LY,0,0,OY)
      CALL TRNSP(TMP4,TMP1,OY,OY)
```

```
      CALL ADD(TMP1,TMP4,SDYY,OY,OY,0)
      GO TO 1000
205   IF(OX.EQ.0) GO TO 4
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL ADD(TMP3,ST,TMP1,LY,LY,0)
4     IF(OX.EQ.0) CALL SCM(ST,1.0,TMP1,LY,LY,0)
      CALL MPY(ONEIJ,TMP2,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP1,TMP4,OY,LY,0,0,LY)
      CALL MPYRT(TMP4,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,ONEKL,TMP3,OY,LY,0,0,OY)
      CALL TRNSP(TMP3,TMP1,OY,OY)
      CALL ADD(TMP1,TMP3,SDYY,OY,OY,0)
      GO TO 1000
104   CALL TRNSP(ONEIJ,TMP1,LY,LY)
      CALL ADD(TMP1,ONEIJ,ONEIJJI,LY,LY,0)
      IF(JROW(IMATRIX(IPARM)).EQ.JCOL(IMATRIX(IPARM)))
     +CALL MPY(ONE,ONEIJ,ONEIJJI,LY,LY,2,0,LY)
      GO TO(206,1010,207,1010) IMATRIX(JPARM)-ILSS
206   CALL MPY(ONEKL,TMP2,TMP1,LY,LY,0,0,LY)
      CALL MPY(TMP1,ONEIJJI,TMP3,LY,LY,0,0,LY)
      CALL TRNSP(TMP3,TMP1,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,TMP4,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,SDYY,OY,LY,0,0,OY)
      GO TO 1000
207   CALL MPY(ONEKL,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEIJJI,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
```

```
      CALL MPYRT(TMP1,L,TMP3,OY,LY,0,0,OY)
      CALL TRNSP(TMP3,TMP1,OY,OY)
      CALL ADD(TMP1,TMP3,SDYY,OY,OY,0)
      GO TO 1000
105   GO TO 1010
106   GO TO 1010
107   CALL TRNSP(ONEIJ,TMP1,LX,LX)
      CALL ADD(TMP1,ONEIJ,ONEIJJI,LX,LX,0)
      IF(JROW(IMATRIX(IPARM)).EQ.JCOL(IMATRIX(IPARM)))
     +CALL MPY(ONE,ONEIJ,ONEIJJI,LX,LX,2,0,LX)
      GO TO(208,209,210,1010,1010,1010,1010)IMATRIX(JPARM)-ILSS
208   CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEKL,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,ONEIJJI,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,SDYX,OY,LX,0,0,OX)
      CALL MPY(ONEKL,TMP2,TMP1,LY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,LY,LY,0,0,LX)
      CALL MPY(TMP3,ONEIJJI,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL TRNSP(TMP3,TMP1,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,TMP4,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,SDYY,OY,LY,0,0,OY)
      GO TO 1000
209   CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEKL,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,ONEIJJI,TMP1,OY,LX,0,0,LX)
```

```
      CALL MPYRT(TMP1,G,SDYX,OY,LX,0,0,OX)
      CALL MPY(ONEKL,ONEIJJI,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL TRNSP(TMP3,TMP1,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,TMP4,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,SDYY,OY,LY,0,0,OY)
      GO TO 1000
210   CALL MPY(ONEKL,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,ONEIJJI,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,SDYX,OY,LX,0,0,OX)
      CALL MPY(TMP2,B,TMP1,LY,LY,0,0,LX)
      CALL MPY(TMP1,ONEIJJI,TMP3,LY,LX,0,0,LX)
      CALL MPYRT(TMP3,B,TMP1,LY,LX,0,0,LY)
      CALL MPYRT(TMP1,TMP2,TMP3,LY,LY,0,0,LY)
      CALL MPY(ONEKL,TMP3,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,TMP4,OY,LY,0,0,OY)
      CALL MPY(L,TMP3,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,ONEKL,TMP3,OY,LY,0,0,OY)
      CALL ADD(TMP3,TMP4,SDYY,OY,OY,0)
      GO TO 1000
108   GO TO(211,212,213,1010,214,1010,1010,215)IMATRIX(JPARM)-ILSS
211   CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEKL,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,SDYX,OY,LX,0,0,OX)
```

```
      GO TO 1000
212   CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEKL,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,SDYX,OY,LX,0,0,OX)
      GO TO 1000
213   CALL MPY(ONEKL,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,SDYX,OY,LX,0,0,OX)
      GO TO 1000
214   CALL TRNSP(ONEKL,TMP1,LX,LX)
      CALL ADD(TMP1,ONEKL,TMP3,LX,LX,0)
      IF(JROW(IMATRIX(JPARM)).EQ.JCOL(IMATRIX(JPARM)))
     +CALL MPY(ONE,ONEKL,TMP3,LX,LX,2,0,LX)
      CALL MPY(G,TMP3,TMP1,OX,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,TMP4,OX,LX,0,0,OX)
      CALL TRNSP(TMP4,TMP1,OX,OX)
      CALL ADD(TMP1,TMP4,SDXX,OX,OX,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP4,OY,LY,0,0,LX)
      CALL MPY(TMP4,TMP3,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,SDYX,OY,LX,0,0,OX)
      GO TO 1000
215   CALL MPY(ONEKL,SX,TMP1,OX,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,TMP3,OX,LX,0,0,OX)
      CALL TRNSP(TMP3,TMP1,OX,OX)
      CALL ADD(TMP1,TMP3,SDXX,OX,OX,0)
      GO TO 1000
      END
      SUBROUTINE SND2(IPARM,JPARM,TRSND)
```

```
      RETURN
      END
      SUBROUTINE FST1(IPARM,FDXX,FDYY,FDYX)
      INTEGER OX,OY
      REAL L,LB
      COMMON /BLK1/A(200,1),B(200,1),L(200,1),ST(200,1),
     +PE(200,1),PW(200,1),SX(200,1),G(200,1),
     +CA(200,1),CB(200,1),CL(200,1),CST(200,1),
     +CPE(200,1),CPW(200,1),CSX(200,1),CG(200,1)
      COMMON /BLK3/ LX,LY
      COMMON /BLK5/ IMATRIX(80),ICOL(80),IROW(80),IP(80),
     +JROW(16),JCOL(16),OX,OY,M,N
      COMMON /BLK10/ TMP1(1500,1),TMP2(1500,1),TMP3(500,1),
     +TMP4(500,1),TMP5(500,1),TMP6(500,1)
      DIMENSION FDXX(1),FDYY(1),FDYX(1)
      DIMENSION ONE(1,1),ONEIJ(1,1)
      EQUIVALENCE (ONE,TMP5),(ONEIJ,TMP6)
      IF(OX.EQ.0) GO TO 1
      CALL GEN(FDXX,0.0,OX,OX,0)
      CALL GEN(FDYX,0.0,OY,OX,0)
1     CONTINUE
      CALL GEN(FDYY,0.0,OY,OY,0)
      CALL SCMA(TMP1,1.0,LY,0)
      CALL SUB(TMP1,A,TMP2,LY,LY,0)
      CALL INVS(TMP2,LY,DET,TMP5,TMP6)
      CALL GEN(ONEIJ,0.0,JROW(IMATRIX(IPARM)),JCOL(IMATRIX(IPARM)),0)
      ILOC=(ICOL(IPARM)-1)*JROW(IMATRIX(IPARM))+IROW(IPARM)
      ONEIJ(ILOC)=1.0
      LMAX=OX
      IF(OX.LT.OY) LMAX=OY
      CALL SCMA(ONE,1.0,LMAX,2)
```

```
      GO TO(101,102,103,104,107,105,106,108) IMATRIX(IPARM)
101   IF(OX.EQ.0) GO TO 2
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEIJ,TMP3,OY,LY,0,0,LY)
      CALL MPY(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL ADD(TMP3,ST,TMP1,LY,LY,0)
2     IF(OX.EQ.0) CALL SCM(ST,1.0,TMP1,LY,LY,0)
      CALL MPYRT(TMP1,TMP2,TMP3,LY,LY,0,0,LY)
      CALL MPYRT(TMP3,ONEIJ,TMP1,LY,LY,0,0,LY)
      CALL TRNSP(TMP1,TMP3,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,TMP4,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      RETURN
102   CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEIJ,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPY(ONEIJ,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL TRNSP(TMP3,TMP1,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(TMP2,TMP4,TMP1,LY,LY,0,0,LY)
      CALL MPY(L,TMP1,TMP3,OY,LY,0,0,LY)
```

```
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      RETURN
103   IF(OX.EQ.0) GO TO 10
      CALL MPY(ONEIJ,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL ADD(TMP3,ST,TMP1,LY,LY,0)
10    IF(OX.EQ.0) CALL SCM(ST,1.0,TMP1,LY,LY,0)
      CALL MPY(TMP2,TMP1,TMP3,LY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,LY,LY,0,0,LY)
      CALL MPY(ONEIJ,TMP1,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,L,TMP1,OY,LY,0,0,OY)
      CALL TRNSP(TMP1,TMP3,OY,OY)
      CALL ADD(TMP1,TMP3,FDYY,OY,OY,0)
      RETURN
104   CALL TRNSP(ONEIJ,TMP1,LY,LY)
      CALL ADD(ONEIJ,TMP1,TMP3,LY,LY,0)
      IF(JROW(IMATRIX(IPARM)).EQ.JCOL(IMATRIX(IPARM)))
     +CALL MPY(ONE,ONEIJ,TMP3,LY,LY,2,0,LY)
      CALL MPY(TMP2,TMP3,TMP1,LY,LY,0,0,LY)
      CALL MPY(L,TMP1,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      RETURN
105   CALL MPY(ONE,ONEIJ,FDYY,OY,OY,2,0,OY)
      RETURN
106   CALL MPY(ONE,ONEIJ,FDXX,OX,OX,2,0,OX)
```

```
      RETURN
107   CALL TRNSP(ONEIJ,TMP1,LX,LX)
      CALL ADD(TMP1,ONEIJ,TMP3,LX,LX,0)
      IF(JROW(IMATRIX(IPARM)).EQ.JCOL(IMATRIX(IPARM)))
     +CALL MPY(ONE,ONEIJ,TMP3,LX,LX,2,0,LX)
      CALL MPY(G,TMP3,TMP1,OX,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDXX,OX,LX,0,0,OX)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP4,OY,LY,0,0,LX)
      CALL MPY(TMP4,TMP3,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPYRT(TMP1,B,TMP3,OY,LX,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      RETURN
108   CALL MPY(G,SX,TMP1,OX,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,TMP3,OX,LX,0,0,OX)
      CALL TRNSP(TMP3,TMP1,OX,OX)
      CALL ADD(TMP1,TMP3,FDXX,OX,OX,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,FDYX,OY,LX,0,0,OX)
      RETURN
      END
      SUBROUTINE FST2(IPARM,FDXX,FDYY,FDYX)
      INTEGER OX,OY
      REAL L,LB
      COMMON /BLK2/A(200,1),B(200,1),L(200,1),ST(200,1),
     +PE(200,1),PW(200,1),SX(200,1),G(200,1),
     +CA(200,1),CB(200,1),CL(200,1),CST(200,1),
```

```
+CPE(200,1),CPW(200,1),CSX(200,1),CG(200,1)
 COMMON /BLK4/ LX,LY
 COMMON /BLK5/ IMATRIX(80),ICOL(80),IROW(80),IP(80),
+JROW(16),JCOL(16),OX,OY,M,N
 COMMON /BLK10/ TMP1(1500,1),TMP2(1500,1),TMP3(500,1),
+TMP4(500,1),TMP5(500,1),TMP6(500,1)
 DIMENSION FDXX(1),FDYY(1),FDYX(1)
 DIMENSION ONE(1,1),ONEIJ(1,1)
 EQUIVALENCE (ONE,TMP5),(ONEIJ,TMP6)
 IF(OX.EQ.0) GO TO 1
 CALL GEN(FDXX,0.0,OX,OX,0)
 CALL GEN(FDYX,0.0,OY,OX,0)
1      CONTINUE
 CALL GEN(FDYY,0.0,OY,OY,0)
 CALL SCMA(TMP1,1.0,LY,0)
 CALL SUB(TMP1,A,TMP2,LY,LY,0)
 CALL INVS(TMP2,LY,DET,TMP5,TMP6)
 CALL GEN(ONEIJ,0.0,JROW(IMATRIX(IPARM)),JCOL(IMATRIX(IPARM)),0)
 ILOC=(ICOL(IPARM)-1)*JROW(IMATRIX(IPARM))+IROW(IPARM)
 ONEIJ(ILOC)=1.0
 LMAX=OX
 IF(OX.LT.OY) LMAX=OY
 CALL SCMA(ONE,1.0,LMAX,2)
 KPARM=IMATRIX(IPARM)-8
 GO TO(101,102,103,104,107,105,106,108) KPARM
101    IF(OX.EQ.0) GO TO 2
 CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
 CALL MPY(TMP1,ONEIJ,TMP3,OY,LY,0,0,LY)
 CALL MPY(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
 CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
 CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
```

```
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL ADD(TMP3,ST,TMP1,LY,LY,0)
2     IF(OX.EQ.0) CALL SCM(ST,1.0,TMP1,LY,LY,0)
      CALL MPYRT(TMP1,TMP2,TMP3,LY,LY,0,0,LY)
      CALL MPYRT(TMP3,ONEIJ,TMP1,LY,LY,0,0,LY)
      CALL TRNSP(TMP1,TMP3,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,TMP4,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      GO TO 110
102   CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,ONEIJ,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPY(ONEIJ,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL TRNSP(TMP3,TMP1,LY,LY)
      CALL ADD(TMP1,TMP3,TMP4,LY,LY,0)
      CALL MPY(TMP2,TMP4,TMP1,LY,LY,0,0,LY)
      CALL MPY(L,TMP1,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      GO TO 110
103   IF(OX.EQ.0) GO TO 10
      CALL MPY(ONEIJ,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
```

```
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPY(B,SX,TMP1,LY,LX,0,0,LX)
      CALL MPYRT(TMP1,B,TMP3,LY,LX,0,0,LY)
      CALL ADD(TMP3,ST,TMP1,LY,LY,0)
10    IF(OX.EQ.0) CALL SCM(ST,1.0,TMP1,LY,LY,0)
      CALL MPY(TMP2,TMP1,TMP3,LY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,LY,LY,0,0,LY)
      CALL MPY(ONEIJ,TMP1,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,L,TMP1,OY,LY,0,0,OY)
      CALL TRNSP(TMP1,TMP3,OY,OY)
      CALL ADD(TMP1,TMP3,FDYY,OY,OY,0)
      GO TO 110
104   CALL TRNSP(ONEIJ,TMP1,LY,LY)
      CALL ADD(ONEIJ,TMP1,TMP3,LY,LY,0)
      IF(JROW(IMATRIX(IPARM)).EQ.JCOL(IMATRIX(IPARM)))
     +CALL MPY(ONE,ONEIJ,TMP3,LY,LY,2,0,LY)
      CALL MPY(TMP2,TMP3,TMP1,LY,LY,0,0,LY)
      CALL MPY(L,TMP1,TMP3,OY,LY,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      GO TO 110
105   CALL MPY(ONE,ONEIJ,FDYY,OY,OY,2,0,OY)
      GO TO 110
106   CALL MPY(ONE,ONEIJ,FDXX,OX,OX,2,0,OX)
      GO TO 110
107   CALL TRNSP(ONEIJ,TMP1,LX,LX)
      CALL ADD(TMP1,ONEIJ,TMP3,LX,LX,0)
      IF(JROW(IMATRIX(IPARM)).EQ.JCOL(IMATRIX(IPARM)))
     +CALL MPY(ONE,ONEIJ,TMP3,LX,LX,2,0,LX)
      CALL MPY(G,TMP3,TMP1,OX,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDXX,OX,LX,0,0,OX)
```

```
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP4,OY,LY,0,0,LX)
      CALL MPY(TMP4,TMP3,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,G,FDYX,OY,LX,0,0,OX)
      CALL MPYRT(TMP1,B,TMP3,OY,LX,0,0,LY)
      CALL MPYRT(TMP3,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPYRT(TMP1,L,FDYY,OY,LY,0,0,OY)
      GO TO 110
108   CALL MPY(G,SX,TMP1,OX,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,TMP3,OX,LX,0,0,OX)
      CALL TRNSP(TMP3,TMP1,OX,OX)
      CALL ADD(TMP1,TMP3,FDXX,OX,OX,0)
      CALL MPY(L,TMP2,TMP1,OY,LY,0,0,LY)
      CALL MPY(TMP1,B,TMP3,OY,LY,0,0,LX)
      CALL MPY(TMP3,SX,TMP1,OY,LX,0,0,LX)
      CALL MPYRT(TMP1,ONEIJ,FDYX,OY,LX,0,0,OX)
110   ANUM=N
      CALL SCM(FDYY,ANUM,FDYY,OY,OY,0)
      IF(OX.EQ.0) RETURN
      CALL SCM(FDXX,ANUM,FDXX,OX,OX,0)
      CALL SCM(FDYX,ANUM,FDYX,OY,OX,0)
      RETURN
      END
      SUBROUTINE TRACE(A,C,N,MS)
C
C         CALL TRACE(A,C,N,MS)
C
C         A     AN N BY N MATRIX
C         C     SUM OF DIAGONAL ELEMENTS
C         N     NUMBER OF ROWS OR COLUMNS IN A
C         MS    MODE OF STORAGE OF A
C
```

```
      DIMENSION A(1)
      C=0.
      IF (MS-1) 10,20,30
   10 J=N*N
      K=N+1
      DO 11 I=1,J,K
   11 C=C+A(I)
      RETURN
   20 J=0
      DO 21 K=1,N
      J=J+K
   21 C=C+A(J)
      RETURN
   30 DO 31 I=1,N
   31 C=C+A(I)
      RETURN
      END
      SUBROUTINE READ (A,M,N,MS,NFMT)
CREAD       READ A MATRIX IN UNTRANSPOSED FORM
C      A=MATRIX TO BE READ
C      M=NUMBER OF ROWS IN UNTRANSPOSED MATRIX
C      N=NUMBER OF COLUMNS IN UNTRANSPOSED MATRIX
C      MS=TYPE OF MATRIX
C          0=RECTANGULAR
C          1=PACKED LOWER TRIANGLE
C          2=DIAGONAL (READ AS A VECTOR)
C      NFMT=FORMAT CODE
C          0=PREVIOUS FORMAT CARD DESCRIBES THE MATRIX
C          1=MATRIX PRECEDED IMMEDIATELY BY A FORMAT CARD
      DIMENSION A(1),FMT(65)
      IF(NFMT)14,4,14
C      READ FORMAT
```

```
   14 ICARD=1
      I=1
      J=13
    1 READ(1,2)(FMT(K),K=I,J),KEY
    2 FORMAT(13A6,I2)
      IF(KEY)3,4,3
    3 IF(ICARD.LT.5) GO TO 5
    6 WRITE(2,7)
    7 FORMAT(22H0TOO MANY FORMAT CARDS)
      CALL EXIT
    5 ICARD=ICARD+1
      I=I+13
      J=J+13
      GO TO 1
C      READ MATRIX
    4 IF(MS-1)8,9,10
    8 J=M*N
      ISTEP=M
      K=M
      GO TO 11
    9 J=(M*(M+1))/2
      GO TO 12
   10 J=M
   12 ISTEP=1
      K=1
   11 DO 13 I=1,K
   13 READ(1,FMT)(A(L),L=I,J,ISTEP)
      RETURN
      END
      SUBROUTINE PRINT(B,NUMROW,NUMC,MS,ROW,COL,NPAGE,HEAD,TITLE)
CPRINT    PRINT A MATRIX
```

```
      DIMENSION B(1),HEAD(1),TITLE(6),       FMT(4),FORM(3),FLB(3),
     1ROW(1),COL(1),TEMP(20)
      LOGICAL X,D1,D2
      DATA BLANK/6H      /,FORM(3),FLB(3)/2*1H)/
      DATA TEMP(1)/6H6)    /,TEMP(2)/6H5)    /,TEMP(3)/6H4)    /
      DATA TEMP(4)/6H3)    /,TEMP(5)/6H2)    /,TEMP(6)/6H(I6,2X/
      DATA TEMP(7)/6HA6,2X /,TEMP(8)/6H1P7E15/,TEMP(9)/6H.5)    /
      DATA TEMP(10)/6H(1H09X/,TEMP(11)/6H 7I15)/,TEMP(12)/6H(13X7(/
      DATA TEMP(13)/6H9XA5))/,TEMP(14)/6H10F11./,TEMP(15)/6H(1H012/
      DATA TEMP(16)/6HX10I11/,TEMP(17)/6H(15X10/,TEMP(18)/6H(5XA6)/
      DATA TEMP(19)/6H(I11, /,TEMP(20)/6H5X    /
      IF(NUMROW) 1001,999,1001
  999 WRITE(2,1000)
 1000 FORMAT(29H0M OR N = 0, PRINTING DELETED)
      RETURN
 1001 IF(NUMC) 1002,999,1002
 1002 IF(NUMROW-1000) 1003,1003,1004
 1003 IF(NUMC-1000) 1006,1006,1004
 1004 WRITE(2,1005)
 1005 FORMAT(43H0M OR N GREATER THAN 1000, PRINTING DELETED)
      RETURN
 1006 BIG=ABS (B(1))
      IF(MS-1)3,4,5
    3 J=NUMROW*NUMC
      NUMCOL=NUMC
      GO TO 1
    4 J=(NUMROW*(NUMROW+1))/2
      NUMCOL=NUMROW
      X=.FALSE.
      GO TO 2
    5 J=NUMROW
```

```
      NUMCOL=1
    1 X=.TRUE.
    2 IF(J.LE.1) GO TO 8
  201 DO 202 I=2,J
  202 BIG=AMAX1 (BIG,ABS (B(I)))
    8 CONTINUE
      IF(BIG.NE.0.) GO TO 888
  800 MODE=10
      GO TO 10
  888 MODE=ALOG10(ABS (BIG))
      FMT(1)=TEMP(6)
      FMT(2)=TEMP(7)
      IF(MODE.LE.4.AND.MODE.GE.0) GO TO 12
   10 MAXCOL=7
      FMT(3)=TEMP(8)
      FMT(4)=TEMP(9)
      FORM(1)=TEMP(10)
      FORM(2)=TEMP(11)
      FLB(1)=TEMP(12)
      FLB(2)=TEMP(13)
      GO TO 13
   12 MAXCOL=10
      FMT(3)=TEMP(14)
      FMT(4)=TEMP(MODE+1)
      FORM(1)=TEMP(15)
      FORM(2)=TEMP(16)
      FLB(1)=TEMP(17)
      FLB(2)=TEMP(18)
   13 CONTINUE
      IF (COL.NE.BLANK) GO TO 301
  300 D2=.FALSE.
```

```
      GO TO 302
  301 D2=.TRUE.
  302 IF(ROW.EQ.BLANK) GO TO 303
  304 D1=.TRUE.
      GO TO 15
  303 D1=.FALSE.
   14 CONTINUE
      FMT(1)=TEMP(19)
      FMT(2)=TEMP(20)
   15 CONTINUE
      MCM1=MAXCOL-1
      JCRANK=NUMROW
      LOTP=40
      NLPP=50
      DO 75 K=1,NUMCOL,MAXCOL
      KSTOP= MIN0 (NUMCOL,K+MAXCOL-1)
      KK=(K-1)*NUMROW
      LINE=(-1)
      KSM1=KSTOP-K
      ISTART=1
      IF(X) GO TO 20
   18 JCRANK=1
      JADD=K-1
      JSTART=((JADD*K)/2)+1
   19 ISTART=K
   20 CONTINUE
      DO 75 II=ISTART,NUMROW
      LINE=LINE+1
   29 IF(MOD(LINE,NLPP).NE.0) GO TO 35
   30 CONTINUE
      J=LOTP+NUMROW
```

```
      IF(40.LT.J) GO TO 32
   31 WRITE(2,901)
      WRITE(2,901)
      GO TO 311
   32 CONTINUE
      LOTP=0
      NPAGE=NPAGE+1
      WRITE(2,900)(HEAD(I),I=1,20),(TITLE(I),I=1,6),      NPAGE
  900 FORMAT(1H15X20A6//32X6A10,25X   ,4HPAGEI5)
  311 CONTINUE
      WRITE(2,FORM)(I,I=K,KSTOP)
      IF(D2) GO TO 33
      GO TO 34
   33 CONTINUE
      WRITE(2,FLB)(COL(I),I=K,KSTOP)
   34 CONTINUE
      WRITE(2,901)
  901 FORMAT(1H0)
   35 CONTINUE
      JJ=MCM1
      IF(X) GO TO 50
   40 JSTART=JSTART+JADD
      JSTOP=JSTART+ MIN0 (LINE,MCM1)
      JADD=JADD+1
      GO TO 60
   50 JSTART=II+KK
      JSTOP=JSTART+NUMROW* MIN0 (JJ,KSM1)
   60 CONTINUE
      LOTP=LOTP+1
      IF(D1)  GO TO 70
   65 WRITE(2,FMT)II,(B(J),J=JSTART,JSTOP,JCRANK)
```

```
      GO TO 75
   70 WRITE(2,FMT)II,ROW(II),(B(J),J=JSTART,JSTOP,JCRANK)
   75 CONTINUE
      RETURN
      END
      SUBROUTINE SCMA  (A,C,N,MS)
C     GENERATE SCALAR MATRIX SUBROUTINE
C
C     CALL SCMA(A,C,N,MS)
C
C  A      MATRIX TO BE GENERATED
C
C  C      SCALAR CONSTANT
C
C  N      DIMENSION OF A
C
C  MS     MODE OF STORAGE OF A
C
C     GENERATE A SCALAR MATRIX, A, SUCH THAT A=C * IDENTITY
C
C     NO REPLACEMENTS
C
      DIMENSION A(1)
      IF(MS-1) 10,20,30
   10 L = N*N
      DO 1 J=1,L
    1 A(J) = 0
      K = N + 1
      DO 2 J=1,L,K
    2 A(J) = C
      RETURN
```

```
   20 L = (N*(N+1))/2
      DO21 J=1,L
   21 A(J) = 0
      K = 1
      DO22 J=1,N
      A(K) = C
   22 K = K + J + 1
      RETURN
   30 DO31 J=1,N
   31 A(J) = C
      RETURN
      END
      SUBROUTINE ADD(A,B,C,M,N,MS)
C
C        CALL ADD(A,B,C,M,N,MS)
C
C        A     AN M BY N MATRIX OF MODE OF STORAGE MS
C        B     AN M BY N MATRIX OF MODE OF STORAGE MS
C        C     AN M BY N MATRIX OF MODE OF STORAGE MS
C        M     NUMBER OF ROWS IN A,B,C
C        N     NUMBER OF COLUMNS IN A,B,C
C        MS    MODE OF STORAGE OF A,B,C
C
C        C MAY REPLACE A
C
C        ADD MATRIX A TO MATRIX B.      STORE RESULT IN MATRIX C
C
      DIMENSION A(1),B(1),C(1)
      IF(MS-1) 1,2,3
    1 K=M*N
      GO TO 4
```

```
    2 K=M*(M+1)/2
      GO TO 4
    3 K=M
    4 DO 5 I=1,K
    5 C(I)=A(I)+B(I)
      RETURN
      END
      SUBROUTINE SUB(A,B,C,M,N,MS)
C
C         CALL SUB(A,B,C,M,N,MS)
C
C         A    AN M BY N MATRIX OF MODE OF STORAGE MS
C         B    AN M BY N MATRIX OF MODE OF STORAGE MS
C         C    AN M BY N MATRIX OF MODE OF STORAGE MS
C         M    NUMBER OF ROWS IN A,B,C
C         N    NUMBER OF COLUMNS IN A,B,C
C         MS   MODE OF STORAGE OF A,B,C
C
C         C MAY REPLACE A
C
C         SUBTRACT MATRIX B FROM MATRIX A.  STORE RESULT IN MATRIX C
C
      DIMENSION A(1),B(1),C(1)
      IF(MS-1) 1,2,3
    1 K=M*N
      GO TO 4
    2 K=M*(M+1)/2
      GO TO 4
    3 K=M
    4 DO 5 I=1,K
    5 C(I)=A(I)-B(I)
```

```
      RETURN
      END
      SUBROUTINE SCM(A,C,B,M,N,MS)
C
C        CALL SCM(A,C,B,M,N,MS)
C
C        A    AN M BY N MATRIX OF MODE OF STORAGE MS
C        C    A SCALAR
C        B    AN ARRAY OF AT LEAST THE NUMBER OF LOCATIONS IN A
C        M    NUMBER OF ROWS IN A
C        N    NUMBER OF COLUMNS IN A
C        MS   MODE OF STORAGE OF A
C
C        B MAY REPLACE A
C
C        MULTIPLY EACH ELEMENT OF THE ARRAY A TIMES THE SCALAR C AND
C        STORE THE RESULT IN  B
C
      DIMENSION A(1),B(1)
      IF (MS .EQ. 3) GO TO 199
      IF (MS-1) 99,199,299
99    MN=M*N
      GO TO 499
199   MN=(M*(M+1))/2
      GO TO 499
299   MN=M
499   DO 500 I=1,MN
500   B(I)=C*A(I)
      RETURN
      END
      SUBROUTINE MPYTR(A,B,C,P,Q,MSA,MSB,R)
C
```

```
C         CALL MPYTR(A,B,C,MA,NA,MSA,MSB,NB)
C
C         A    AN MA BY NA ARRAY
C         B    AN MA BY NB ARRAY
C         C    AN ARRAY TO CONTAIN THE MATRIX PRODUCT OF A TRANSPOSE TIM
C         MA   NUMBER OF ROWS IN A
C         NA   NUMBER OF COLUMNS IN A
C         MSA  MODE OF STORAGE OF A
C         MSB  MODE OF STORAGE OF B
C         NB   NUMBER OF COLUMNS OF B
C
C         CALCULATE THE MATRIX PRODUCT OF A TRANSPOSE TIMES B AND STORE
C
C
      INTEGER P,Q,R
      DIMENSION A(P,Q),B(P,R),C(Q,R)
      IF (MSB .EQ. 1) GO TO 200
      IF (MSA .EQ. 3) GO TO 300
      DO 100 I=1,R
      DO 100 J=1,Q
      X=0.
      DO 101 K=1,P
101   X=X+A(K,J)*B(K,I)
100   C(J,I)=X
      RETURN
200   II=0
      DO 205 I=1,R
      DO 201 J=1,Q
      X=0.
      IC=II
      DO 202 M=1,I
```

```
      IC=IC+1
202   X=X+A(M,J)*B(IC,1)
      IF (I .EQ. R) GO TO 299
      L=I+1
      IC=IC+I
      DO 203 M=L,P
      X=X+A(M,J)*B(IC,1)
203   IC=IC+M
299   C(J,I)=X
201   CONTINUE
205   II=II+I
      RETURN
300   II=0
      DO 301 I=1,P
      II=II+I
      DO 301 J=1,R
      IC=II
      X=0.
      DO 302 K=I,Q
      X=X+A(IC,1)*B(K,J)
302   IC=IC+K
301   C(I,J)=X
      RETURN
      END
      SUBROUTINE MPYRT(A,B,C,P,Q,MSA,MSB,R)
C
C         CALL MPYRT(A,B,C,MA,NA,MSA,MSB,MB)
C
C         A    AN MA BY NA ARRAY
C         B    AN MB BY NA ARRAY
C         C    AN ARRAY TO CONTAIN THE MATRIX PRODUCT OF A TIMES B TRANS
C         MA   NUMBER OF ROWS IN A
```

```
C          NA    NUMBER OF COLUMNS IN A
C          MSA   MODE OF STORAGE OF A
C          MSB   MODE OF STORAGE OF B
C          MB    NUMBER OF ROWS IN B
C
C          CALCULATE THE MATRIX PRODUCT OF A TIMES B TRANSPOSE AND STORE
C
      INTEGER P,Q,R
      DIMENSION A(P,Q),B(R,Q),C(P,R)
      IF (MSA .EQ. 1) GO TO 200
      IF (MSB .EQ. 3) GO TO 300
      DO 100 I=1,R
      DO 100 J=1,P
      X=0.
      DO 101 K=1,Q
101   X=X+A(J,K)*B(I,K)
100   C(J,I)=X
      RETURN
200   II=0
      DO 205 I=1,P
      DO 201 J=1,R
      X=0.
      IC=II
      DO 202 M=1,I
      IC=IC+1
202   X=X+B(J,M)*A(IC,1)
      IF (I .EQ. Q) GO TO 299
      L=I+1
      IC=IC+I
      DO 203 M=L,Q
      X=X+B(J,M)*A(IC,1)
203   IC=IC+M
```

```
299   C(I,J)=X
201   CONTINUE
205   II=II+I
      RETURN
300   II=0
      DO 301 I=1,R
      DO 302 J=1,P
      IC=II
      X=0.
      DO 303 K=1,I
      IC=IC+1
303   X=X+A(J,K)*B(IC,1)
302   C(J,I)=X
301   II=II+I
      RETURN
      END
      SUBROUTINE MPY(A,B,C,P,Q,MSA,MSB,R)
C
C        CALL MPY(A,B,C,MA,NA,MSA,MSB,NB)
C
C        A    AN MA BY NA MATRIX
C        B    AN NA BY NB MATRIX
C        C    AN ARRAY TO CONTAIN THE MATRIX PRODUCT OF A TIMES B
C        MA   NUMBER OF ROWS IN A
C        NA   NUMBER OF COLUMNS IN A
C        MSA  MODE OF STORAGE OF A
C        MSB  MODE OF STORAGE OF B
C        NB   NUMBER OF COLUMNS OF B
C
C        CALCULATE THE MATRIX PRODUCT OF A TIMES B AND STORE IN C
C
```

```
      INTEGER P,Q,R,P1
      DIMENSION A(P,Q),B(Q,R),C(P,R)
      I=1+MSA+4*MSB
      GO TO (100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,
     11400,1500,1600),I
100   DO 101 I=1,R
      DO 101 J=1,P
      X=0.
      DO 102 K=1,Q
102   X=X+A(J,K)*B(K,I)
101   C(J,I)=X
      RETURN
200   DO 205 I=1,R
      JJ=0
      DO 201 J=1,P
      X=0.
      JC=JJ
      DO 203 M=1,J
      JC=JC+1
203   X=X+A(JC,1)*B(M,I)
      IF (J .EQ. Q) GO TO 299
      L=J+1
      JC=JC+J
      DO 204 M=L,Q
      X=X+A(JC,1)*B(M,I)
204   JC=JC+M
299   C(J,I)=X
201   JJ=JJ+J
205   CONTINUE
      RETURN
300   DO 301 I=1,R
```

```
      DO 301 J=1,P
301   C(J,I)=B(J,I)*A(J,1)
      RETURN
400   DO 401 I=1,R
      JJ=0
      DO 402 J=1,P
      X=0.
      JC=JJ
      DO 403 M=1,J
      JC=JC+1
403   X=X+A(JC,1)*B(M,I)
      C(J,I)=X
402   JJ=JJ+J
401   CONTINUE
      RETURN
500   II=0
      DO 505 I=1,R
      DO 501 J=1,P
      X=0.
      IC=II
      DO 502 M=1,I
      IC=IC+1
502   X=X+A(J,M)*B(IC,1)
      IF (I .EQ. Q) GO TO 599
      L=I+1
      IC=IC+I
      DO 503 M=L,Q
      X=X+A(J,M)*B(IC,1)
503   IC=IC+M
599   C(J,I)=X
501   CONTINUE
505   II=II+I
```

```
      RETURN
600   II=0
      P1=P-1
      DO 601 I=1,P
      JJ=0
      DO 602 J=1,P
      X=0.
      IC=II
      JC=JJ
      IF (I-J) 625,650,675
625   DO 626 M=1,I
      IC=IC+1
      JC=JC+1
626   X=X+A(IC,1)*B(JC,1)
      L=J-1
      DO 627 M=I,L
      JC=JC+1
      IC=IC+M
627   X=X+A(IC,1)*B(JC,1)
      IF (J .EQ. P) GO TO 699
      DO 629 M=J,P1
      IC=IC+M
      JC=JC+M
629   X=X+A(IC,1)*B(JC,1)
      GO TO 699
675   DO 651 M=1,J

      IC=IC+1
      JC=JC+1
651   X=X+A(IC,1)*B(JC,1)
      L=I-1
```

```
      DO 652 M=J,L
      IC=IC+1
      JC=JC+M
652   X=X+A(IC,1)*B(JC,1)
      IF (I .EQ. P) GO TO 699
      DO 654 M=I,P1
      IC=IC+M
      JC=JC+M
654   X=X+A(IC,1)*B(JC,1)
      GO TO 699
650   DO 676 M=1,I
      IC=IC+1
676   X=X+A(IC,1)*B(IC,1)
      IF (I .EQ. P) GO TO 699
      DO 678 M=I,P1
      IC=IC+M
678   X=X+A(IC,1)*B(IC,1)
699   C(I,J)=X
602   JJ=JJ+J
601   II=II+I
      RETURN
900   DO 901 I=1,Q
      DO 901 J=1,P
901   C(J,I)=A(J,I)*B(I,1)
      RETURN
1100  DO 1101 I=1,P
1101  C(I,1)=A(I,1)*B(I,1)
      RETURN
1300  II=0
      DO 1301 I=1,R
      II=II+I
```

```
      DO 1302 J=1,P
      X=0.
      IC=II
      DO 1303 M=I,Q
      X=X+A(J,M)*B(IC,1)
1303  IC=IC+M
1302  C(J,I)=X
1301  CONTINUE
      RETURN
700   CONTINUE
800   CONTINUE
1000  CONTINUE
1200  CONTINUE
1400  CONTINUE
1500  CONTINUE
1600  CONTINUE
      PAUSE
      RETURN
      END
      SUBROUTINE GEN (A,C,M,N,MS)
C     GENERATION SUBROUTINE
C
C     CALL GEN(A,C,M,N,MS)
C
C   A      MATRIX TO BE GENERATED
C
C   C      SCALAR CONSTANT
C
C   M      NUMBER OF ROWS IN A
C
C   N      NUMBER OF COLUMNS IN A
C
```

```
C  MS     MODE OF STORAGE OF A
C
C    GENERATE A MATRIX, EACH ELEMENT OF WHICH EQUALS C
C
C    NO REPLACEMENTS
C
      DIMENSION A(1)
      IF(MS-1) 10,20,30
   10 L = M*N
      DO 1 J=1,L
    1 A(J) = C
      RETURN
   20 L = (N*(N+1))/2
      DO21 J=1,L
   21 A(J) = C
      RETURN
   30 DO31 J=1,N
   31 A(J) = C
      RETURN
      END
      SUBROUTINE IPSMSL(M,N,A,B,C,EPS,D,E,F,DET,IERR,A1,A2,Y)
C ******* INVERT PARTITIONED SYMMETRIC MATRIX STORED LINEARLY
C                        -1
C           (A        )                (D        )
C           (         ) (INPUT)  =     (         ) (OUTPUT)
C           (B       C)                (E       F)
C
C          WHERE A,B,C,D,E,F ARE LINEARLY STORED MATRICES,AND ONLY THE
C          HALF OF SYMMETRIC MATRICES IS STORED
C           A(MXM), SYMMETRIC              D(MXM), SYMMETRIC
C           B(NXM)                        E(NXM)
C           C(NXN), SYMMETRIC              F(NXN), SYMMETRIC
```

```
C          EPS    IF ANY PIVOTAL ELEMENT IS LESS THAN EPS,THE MATRIX IS
C                 CONSIDERED SINGULAR AND CONTROL IS TRANSFERRED TO THE
C                 PROGRAM WITH IERR=1
C          DET     = DETERMINANT OF PARTITIONED MATRIX TO BE INVERTED
C          IERR    =0 IMPLIES THE MATRIX IS NON-SINGULAR
C          A1     INTERNAL MATRIX,MUST BE DIMENSIONED IN CALLING PROGRA
C                 LEAST N*N
C          A2     INTERNAL MATRIX, MUST BE DIMENSIONED IN CALLING PROGR
C                 LEAST THE LARGER OF N*N OR N*M
C          Y      INTERNAL VECTOR,MUST BE DIMENSIONED IN CALLING PROGRA
C                 LEAST THE LARGER OF M OR N
C         CAUTION  THIS ROUTINE CALLS SUBROUTINES ISMSL AND MMSL
C ************************************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),C(1),D(1),E(1),F(1),A1(1),A2(1),Y(1)
      M2=(M*(M+1))/2
      NN=N*N
      NM=N*M
      CALL ISMSL(N,C,A1,Y,DET1,EPS,IERR)
      IF(IERR.GT.0)RETURN
      CALL MMSL(N,N,M,A1,B,A2,-1,0,1)
      CALL MMSL(M,N,M,B,A2,D,1,0,0)
      DO 1 I=1,M2
      D(I)=A(I)-D(I)
    1 CONTINUE
      CALL ISMSL(M,D,D,Y,DET2,EPS,IERR)
      IF(IERR.GT.0)RETURN
      CALL MMSL(N,M,M,A2,D,E,0,-1,1)
      DO 2 I=1,NM
      E(I)=-E(I)
    2 CONTINUE
```

```
      CALL MMSL(N,M,N,B,E,A2,0,1,1)
      DO 3 I=1,NN
      A2(I)=-A2(I)
    3 CONTINUE
      K=-N
      DO 4 I=1,N
      K=K+N+1
      A2(K) = 1.0 + A2(K)
    4 CONTINUE
      CALL MMSL(N,N,N,A1,A2,F,-1,0,0)
      DET=DET1*DET2
      RETURN
      END
      SUBROUTINE MPSMSL(N,M,A,B,C,D,E,F,X,Y,Z,A1,A2)
C ******** MULTIPLICATION OF PARTITIONED SYMMETRIC MATRICES
C
C            (A    B")(D    E")(A    B")   (X      )
C            (       )(       )(       ) = (       )
C            (B    C )(E    F )(B    C )   (Y    Z )
C
C         WHERE A,B,C,D,E,F,X,Y,Z ARE LINEARLY STORED MATRICES, AND ON
C         THE LOWER HALF OF SYMMETRIC MATRICES IS STORED
C            A(MXM),SYMMETRIC    D(MXM),SYMMETRIC    X(MXM),SYMMETRIC
C            B(NXM)              E(NXM)              Y(NXM)
C            C(NXN),SYMMETRIC    F(NXN),SYMMETRIC    Z(NXN),SYMMETRIC
C         A1,A2    INTERNAL MATRICES STORED LINEARLY,MUST BE DIMENSION
C                  CALLING PROGRAM BY AT LEAST THE LARGER OF N*N OR M*
C         CAUTION  THIS ROUTINE CALLS SUBROUTINE MMSL
C ************************************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),C(1),D(1),E(1),F(1),X(1),Y(1),Z(1),A1(1),
```

```
 1A2(1)
  MM=M*M
  M2=(M*(M+1))/2
  MN=M*N
  N2=(N*(N+1))/2
  NN=N*N
  CALL MMSL(M,M,M,A,D,A1,-1,-1,1)
  CALL MMSL(M,N,M,B,E,A2,1,0,1)
  DO 1 I=1,MM
  A1(I)=A1(I)+A2(I)
1 CONTINUE
  CALL MMSL(M,M,M,A1,A,X,0,-1,0)
  CALL MMSL(N,M,M,B,A1,Y,0,1,1)
  CALL MMSL(M,M,N,A,E,A1,-1,1,1)
  CALL MMSL(M,N,N,B,F,A2,1,-1,1)
  DO 2 I=1,MN
  A1(I)=A1(I)+A2(I)
2 CONTINUE
  CALL MMSL(M,N,M,A1,B,A2,0,0,0)
  DO 3 I=1,M2
  X(I)=X(I)+A2(I)
3 CONTINUE
  CALL MMSL(N,N,M,C,A1,A2,-1,1,1)
  DO 4 I=1,MN
  Y(I)=Y(I)+A2(I)
4 CONTINUE
  CALL MMSL(N,M,M,B,D,A1,0,-1,1)
  CALL MMSL(N,N,M,C,E,A2,-1,0,1)
  DO 5 I=1,MN
  A1(I)=A1(I)+A2(I)
5 CONTINUE
```

```
      CALL MMSL(N,M,N,A1,B,Z,0,1,0)
      CALL MMSL(N,M,N,B,E,A1,0,1,1)
      CALL MMSL(N,N,N,C,F,A2,-1,-1,1)
      DO 6 I=1,NN
      A1(I)=A1(I)+A2(I)
    6 CONTINUE
      CALL MMSL(N,N,N,A1,C,A2,0,-1,0)
      DO 7 I=1,N2
      Z(I)=Z(I)+A2(I)
    7 CONTINUE
      RETURN
      END
      SUBROUTINE ISMSL(N,A,B,Y,D,EPS,IERR)
C ***** INVERT SYMMETRIC MATRIX STORED LINEARLY
C        N    ORDER OF MATRIX
C        A    MATRIX TO BE INVERTED,STORED LINEARLY,MUST BE GRAMIAN
C        B    A INVERSE,STORED AS A VECTOR
C        Y    INTERNAL DUMMY ARRAY,MUST BE DIMENSIONED IN CALLING PROGR
C        D    DETERMINANT(A)
C        EPS  IF ANY PIVOTAL ELEMENT IS LESS THAN EPS,A IS CONSIDERED S
C             AND CONTROL IS TRANSFERRED TO THE CALLING PROGRAM WITH IE
C        IERR =0 IMPLIES A IS NON-SINGULAR
C ********************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),Y(1)
      IERR=0
      NN=(N*(N+1))/2
      DO 5 I=1,NN
    5 B(I)=A(I)
      D = 1.0
      IF(N.EQ.1) GO TO 260
```

```
      DO 240 L=1,N
      F=B(1)
      IF(F.LT.EPS) GO TO 700
      D=D*F
      F = 1.0/F
      NA=1
      DO 210 I=1,N
      NA=NA+I-1
  210 Y(I)=B(NA)
      NA=0
      NB=1
      DO 220 I=2,N
      NB=NB+1
      H=Y(I)*F
      DO 220 J=2,I
      NB=NB+1
      NA=NA+1
  220 B(NA)=B(NB)-Y(J)*H
      DO 230 J=2,N
      NA=NA+1
  230 B(NA)=-Y(J)*F
  240 B(NN)=-F
      DO 250 I=1,NN
  250 B(I)=-B(I)
      RETURN
  260 L=1
      F=B(1)
      IF(F.LT.EPS) GO TO 700
      B(1) = 1.0/F
      D=F
      RETURN
```

```
700 WRITE (2,1) L,F,D
    IERR=1
    RETURN
  1 FORMAT(1H0,10X,35HTHE MATRIX IS NOT POSITIVE DEFINITE  ,I4,2E11.4)
    END
    SUBROUTINE MMSL(M,K,N,A,B,C,IA,IB,IC)
C ***** MULTIPLY MATRICES STORED LINEARLY
C       M,K,N ORDER OF MATRICES,I.E. A(MXK),B(KXN),C(MXN)
C       A,B   MATRICES TO BE MULTIPLIED,STORED AS VECTORS
C       C     AB  (A,B,C MUST BE DIMENSIONED IN CALLING PROGRAM)
C       IA,IB IA=0 , IB=0  IMPLIES  C=AB
C             IA=1 , IB=0  IMPLIES  C=A"B
C             IA=0 , IB=1  IMPLIES  C=AB"
C             IA=1 , IB=1  IMPLIES  C=A"B"
C             IA=-1  IMPLIES A IS SYMMETRIC,ONLY LOWER HALF IS STORED
C             IB=-1  IMPLIES B IS SYMMETRIC,ONLY LOWER HALF IS STORED
C       IC    IC=0   IMPLIES C IS SYMMETRIC,ONLY LOWER HALF IS STORED
C             OTHERWISE C IS A FULL MATRIX
C *************************
    LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
    DIMENSION A(1),B(1),C(1)
    NN=N
    DO 17 I=1,M
    II=(I-1)*N
    IK=(I-1)*K
    I1=(I*(I-1))/2
    IF(IC.EQ.0)NN=I
    DO 17 J=1,NN
    JJ=(J-1)*K
    J1=(J*(J-1))/2
    VW = 0.0
```

```
      DO 12 L=1,K
      IF(IA)3,2,1
    1 LI=(L-1)*M+I
      GO TO 6
    2 LI=IK+L
      GO TO 6
    3 IF(I-L)5,4,4
    4 LI=I1+L
      GO TO 6
    5 LI=(L*(L-1))/2+I
    6 IF(IB)9,8,7
    7 IL=JJ+L
      GO TO 12
    8 IL=(L-1)*N+J
      GO TO 12
    9 IF(L-J)11,10,10
   10 IL=(L*(L-1))/2+J
      GO TO 12
   11 IL=J1+L
   12 VW=VW+A(LI)*B(IL)
      IF(IC)13,14,13
   13 IJ=II+J
      GO TO 17
   14 IF(I-J)15,16,16
   15 IJ=J1+I
      GO TO 17
   16 IJ=I1+J
   17 C(IJ)=VW
      RETURN
      END
      SUBROUTINE TRNSP(A,B,IR,IC)
```

```
      DIMENSION A(1),B(1)
      DO 1 J=1,IC
      DO 1 I=1,IR
      ILOC=(I-1)*IR+J
      JLOC=(J-1)*IC+I
1     B(JLOC)=A(ILOC)
      RETURN
      END
      SUBROUTINE INVS(A,N,C,W1,W2)
C
C      CALL INV(A,N,C,W1,W2)
C
C           A      MATRIX TO BE INVERTED
C           N      NUMBER OF COLUMNS(ROWS) OF A
C           C      DETERMINANT OF A STORED HERE
C           W1     N LOCATIONS OF WORKING STORAGE
C           W2     N LOCATIONS OF WORKING STORAGE
C
C      INVERT AN N BY N NON-SINGULAR MATRIX A IN PLACE.  LEAVE
C           THE DETERMINANT OF A IN C.
C
C      INVERSE REPLACES A
C
      DIMENSION W1(N), W2(N), A(N,N)
      INTEGER W1,W2
      INTEGER FLIPS
      EQUIVALENCE (M,T,ISAV),(L,U,JSAV)
      FLIPS=0
C      D=1.0D0
      D=1.000
      DO 1 K=1,N
```

```
      ISAV=K
      JSAV=K
      X=0
      DO 3 I=K,N
      DO 3 J=K,N
      IF (X-ABS(A(I,J))) 4,3,3
    4 X=ABS(A(I,J))
      ISAV=I
      JSAV=J
    3 CONTINUE
   71 IF(ISAV-K)  7,6,7
    7 DO 8 J=1,N
      X=A(K,J)
      A(K,J)=A(ISAV,J)
    8 A(ISAV,J)=X
    6 IF(JSAV-K)  11,10,11
   11 DO 12 I=1,N
      X=A(I,K)
      A(I,K)=A(I,JSAV)
   12 A(I,JSAV)=X
   10 Y=A(K,K)
      W1(K)=ISAV
      W2(K)=JSAV
      FLIPS=FLIPS+ISAV-K+JSAV-K
      D=Y*D
      IF(Y) 70,777,70
   70 A(K,K)=1.0D0/Y
      DO 50 I=1,N
      IF(I-K)14,50,14
   14 Z=A(I,K)
      Z=-Z/Y
```

```
      A(I,K)=Z
      DO 18 J=1,N
      IF(J-K)17,18,17
   17 U=A(K,J)
      T=A(I,J)
      A(I,J)=T+U*Z
   18 CONTINUE
   50 CONTINUE
      DO 19 J=1,N
      IF(J-K)20,19,20
   20 Z=A(K,J)
      A(K,J)=Z/Y
   19 CONTINUE
    1 CONTINUE
      DO 26 L=1,N
      K=N-L+1
      IF(W2(K)-K)  23,22,23
   23 M=W2(K)
      DO 24 J=1,N
      X=A(K,J)
      A(K,J)=A(M,J)
   24 A(M,J)=X
   22 IF(W1(K)-K)  27,26,27
   27 M=W1(K)
      DO 28 I=1,N
      X=A(I,K)
      A(I,K)=A(I,M)
   28 A(I,M)=X
   26 CONTINUE
      FLIPS=FLIPS-2*(FLIPS/2)
      IF(FLIPS)  500,777,500
```

```
500 D=-D
777 C=D
    RETURN
    END
```

Appendix F

Listing of Estimation Program

```
      PROGRAM LISREL(INPUT=500,OUTPUT=500,PUNCH=500,TAPE5=INPUT,
     +TAPE6=OUTPUT,TAPE7=PUNCH,TAPE9=500)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      COMMON/DISP/SYY(120), SXY(225),SXX(120),CONST,MV(80),
     +SYY10(120),SYX10(225),SXX10(120)
       COMMON/ORDR/P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
     +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      COMMON/SOL/AY(225),AX(225),B(225),G(225),R(120),
     1 V(120),TE(225),TD(225),
     +AY10(225),AX10(225),B10(225),G10(225),R10(120),V10(120),
     +TE10(225),TD10(225)
      COMMON/OLP/OAY(225),OAX(225),OB(225),OG(225),OR(120),
     1OV(120),OTE(225),OTD(225),
     +OAY10(225),OAX10(225),OB10(225),OG10(225),OR10(120),
     +OV10(120),OTE10(225),OTD10(225)
      COMMON/TID/SEC
      COMMON/PRNT/IPF
      DIMENSION HEAD(20),X(80),Y(80),GG(80),H(80),D(80),E(3240),FMT(10)
      DATA HALT/4HSTOP/,FMT/20H(5X,I5,5X,10F11.3)  /
      CALL ERRSET (208,256,-1,1)
C      CALL PDUMP(66507B,66510B,4)
      IC=80
    5 READ(5,100)HEAD
      IF(HEAD(1).EQ.HALT)CALL EXIT
      WRITE(6,200)HEAD
      READ (5,300)P,Q,M,M10,N,N10,NP,NP10,SEC,OX,OZ,OE,OA,OO,OS,OSC,IS,
     +INDS,INDO
      LID=INDO
      O1=MOD(LID,2).EQ.1
```

```
      LID=LID/2
      O2=MOD(LID,2).EQ.1
      LID=LID/2
      O3=MOD(LID,2).NE.1
      WRITE(6,400)P,Q,M,M10,N,N10,NP,NP10,OX,OZ,OE,OA,OO,OS,OSC,IS,INDS,
     +INDO,SEC
      P2=(P*(P+1))/2
      Q2=(Q*(Q+1))/2
      PQ=P*Q
      MP=P*M
      N2=(N*(N+1))/2
      MM=M*M
      NQ=Q*N
      M2=(M*(M+1))/2
      MN=M*N
      MP10=P*M10
      N20=(N10*(N10+1))/2
      MM10=M10*M10
      NQ10=Q*N10
      M20=(M10*(M10+1))/2
      MN10=M10*N10
      CALL INIT(FMT,NP,NP10,IERR,E(1),E(1621),OS)
      IF (IERR.EQ.0) GO TO 10
      WRITE (6,500)
      GO TO 5
   10 IF(.NOT.O1)GO TO 15
      WRITE(6,600)
      WRITE(6,601)
  601 FORMAT(* WITHIN GROUPS PARAMETERS*)
      CALL SIDA(AY,AX,B,G,R,V,TE,TD,FMT,P,Q,M,N)
      WRITE(6,602)
  602 FORMAT(* BETWEEN GROUPS PARAMETERS*)
```

```
      CALL SIDA(AY10,AX10,B10,G10,R10,V10,TE10,TD10,FMT,P,Q,
     +M10,N10)
   15 IF(NOF.LE.IC)GO TO 20
      WRITE(6,700)
      GO TO 5
20    CALL MOVE(AY,AX,B,G,R,V,TE,TD,AY10,AX10,B10,G10,R10,V10,TE10,TD10,
     +X,1)
      IF(NOI.LT.NOF)CALL DADDY(MV,X,2)
   25 IPF=0
      SEC=SEC*1.E+6
      T1 = SECOND(VAD)
      CALL STEDE(NOI,X,Y,GG,H,D,F,O3,IERR,NP,NP10)
      IF(IERR.NE.4)GO TO 45
      L=0
      DO 35 I=1,NOI
      DO 30 J=1,I
      L=L+1
   30 E(L) = 0.0
   35 E(L) = 1.0
      CALL FLEPOW(NOI,X,Y,GG,H,D,E,F,O3,IERR,OA,NP,NP10)
   45 T2 = SECOND(VAD)
      T3=(T1-T2)*1.E-6
      WRITE(6,800)T3,IERR
      WRITE(6,601)
      CALL SIDA(AY,AX,B,G,R,V,TE,TD,FMT,P,Q,M,N)
      WRITE(6,602)
      CALL SIDA(AY10,AX10,B10,G10,R10,V10,TE10,TD10,FMT,P,Q,
     +M10,N10)
      IF(IERR.EQ.0)GO TO 50
      OO=.TRUE.
   50 CALL FINOT(X,F,GG,FMT,OSC,NP,NP10,CONST)
```

```
      GO TO 5
100 FORMAT(20A4)
200 FORMAT(*1     LINEAR STRUCTURAL RELATIONSHIPS*//10X,20A4)
300 FORMAT(8I3,1X,F10.0,5X,7L1,3X,3I1)
400 FORMAT(1H0,10X,*P =*,I3/1H0,10X,*Q =*,I3/1H0,10X,*M =*,I3/
   +,1H0,10X,*M2= *,I3/1H0,10X,*N= *,I3/1H0,10X,*N2= *,I3/
   A1H0,10X,*NP =*,I3/1H0,10X,*NP2 =*,I3/1H0,10X,*LOGICAL INDICATORS
   +=*
   B,7L1/1H0,10X,*INTEGER INDICATORS = *,3I1/1H0,10X,*ESTIMATED TIME I
   CN SECONDS =*,F5.0)
500 FORMAT(1H0,*MATRIX TO BE ANALYZED IS NOT POSITIVE DEFINITE*)
600 FORMAT(1H1,5X,*INITIAL SOLUTION*)
700 FORMAT(1H0,*NUMBER OF NON-FIXED PARAMETERS IS GREATER THAN 80*)
800 FORMAT(1H0,10X,*TIME =*,F8.2/1H1,10X,*MAXIMUM LIKELIHOOD SOLUTION*
   A/1H0,10X,*IND =*,I5)
      END
      SUBROUTINE INIT(FMT,NP,NP10,IERR,E,E10,OS)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      COMMON/DISP/SYY(120), SXY(225),SXX(120),CONST,MV(80),
   +SYY10(120),SXY10(225),SXX10(120)
      COMMON /SIG/CYY(120),CXY(225),CXX(120),C(120),D(225),
   +CYY10(120),CXY10(225),CXX10(120),C10(120),D10(225)
      COMMON/SOL/AY(225),AX(225),B(225),G(225),R(120),
   1 V(120),TE(225),TD(225),
   +AY10(225),AX10(225),B10(225),G10(225),R10(120),V10(120),
   +TE10(225),TD10(225)
       COMMON/ORDR/P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
   +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      DIMENSION FMT(1),E(1),E10(1)
```

```
      INME=10HINIT
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
C *** DEFINE DATA MATRIX TO BE ANALYZED AND CONST
      CALL REX(INDS,IS,O1,OX,NP,FMT,V,V10,R,R10,E,E10,OS)
      IF(OX) GO TO 5
      IPNUM=P*(P+1)/2
      DO 1 I=1,IPNUM
      SYY(I)=SYY(I)*NP/(NP-1)
1     CONTINUE
      CALL ISMSL(P,SYY,CYY,TE,DET1,5.E-12,IERR)
      IF(IERR.GT.0) RETURN
      CALL ISMSL(P,SYY10,CYY10,TE10,DET2,5.E-12,IERR)
      DO 2 I=1,IPNUM
2     SYY(I)=SYY(I)*(NP-1)/NP
      GO TO 10
5     IPNUM=P*(P+1)/2
      IQNUM=Q*(Q+1)/2
      IPQNUM=P*Q
      DO 6 I=1,IPNUM
      SYY(I)=SYY(I)*NP/(NP-1)
6     CONTINUE
      DO 7 I=1,IQNUM
      SXX(I)=SXX(I)*NP/(NP-1)
7     CONTINUE
      DO 8 I=1,IPQNUM
      SXY(I)=SXY(I)*NP/(NP-1)
8     CONTINUE
      CALL IPSMSL(P,Q,SYY,SXY,SXX,5.E-12,CYY,CXY,CXX,DET1,IERR,B,G,AX)
      IF (IERR.GT.0) RETURN
```

```
      CALL IPSMSL(P,Q,SYY10,SXY10,SXX10,5.E-12,CYY10,CXY10,CXX10,
     +DET2,IERR,B10,G10,AX10)
      DO 9 I=1,IPNUM
9     SYY(I)=SYY(I)*(NP-1)/NP
      DO 11 I=1,IQNUM
11    SXX(I)=SXX(I)*(NP-1)/NP
      DO 12 I=1,IPQNUM
12    SXY(I)=SXY(I)*(NP-1)/NP
10    IF(IERR.GT.0) RETURN
      CONST=NP10*(NP-1)*ALOG(DET1)+NP10*ALOG(DET2)+(P+Q)*NP*NP10
C *** DESCRIBE MODEL (LOCATION OF FREE, FIXED, CONSTRAINED PARAMETERS)
      CALL PACK(MV)
C *** READ IN STARTING POINT
      CALL STRTVL(AY,AX,B,G,R,V,TE,TD,P,Q,M,N,MP,NQ,MM,MN,N2,M2)
      CALL STRTVL(AY10,AX10,B10,G10,R10,V10,TE10,TD10,P,Q,M10,N10,
     +MP10,NQ10,MM10,MN10,N20,M20)
      RETURN
      END
      SUBROUTINE SELECT(S,S10,OS)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL OS
      INTEGER P,Q,P2,Q2,PQ
      COMMON/DISP/SYY(120), SXY(225),SXX(120),CONST,MV(80),
     +SYY10(120),SXY10(225),SXX10(120)
       COMMON/ORDR/P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
     +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      DIMENSION S(1),MS(30),S10(1)
      INME=10HSELECT
C     WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
```

```
      IF(.NOT.OS)GO TO 35
C ******* SELECTION OF VARIABLES
      READ(5,100)P,Q
      MAB=P+Q
      READ(5,200)(MS(I),I=1,MAB)
      I1=0
      MA1=P+1
      DO 10 I=1,P
      DO 10 J=1,I
      LI=MS(I)
      I1=I1+1
      LJ=MS(J)
      IF(LI.GE.LJ)GO TO 5
      LI=LJ
      LJ=MS(I)
    5 IJ=((LI-1)*LI)/2+LJ
      SYY(I1)=S(IJ)
      SYY10(I1)=S10(IJ)
   10 CONTINUE
      IF(Q.EQ.0)GO TO 32
      I1=0
      DO 20 I=MA1,MAB
      DO 20 J=1,P
      LI=MS(I)
      I1=I1+1
      LJ=MS(J)
      IF(LI.GE.LJ)GO TO 15
      LI=LJ
      LJ=MS(I)
   15 IJ=((LI-1)*LI)/2+LJ
      SXY(I1)=S(IJ)
```

```
      SXY10(I1)=S10(IJ)
   20 CONTINUE
      I1=0
      DO 30 I=MA1,MAB
      DO 30 J=MA1,I
      LI=MS(I)
      I1=I1+1
      LJ=MS(J)
      IF(LI.GE.LJ)GO TO 25
      LI=LJ
      LJ=MS(I)
   25 IJ=((LI-1)*LI)/2+LJ
      SXX(I1)=S(IJ)
      SXX10(I1)=S10(IJ)
   30 CONTINUE
   32 P2=(P*MA1)/2
      Q2=(Q*(Q+1))/2
      PQ=P*Q
      MP=P*M
      NQ=N*Q
      MP10=P*M10
      NQ10=Q*N10
C ******* PARTITION S INTO THE PARTS SYY,SXY,SXX
      RETURN
   35 DO 40 I=1,P2
      SYY(I)=S(I)
      SYY10(I)=S10(I)
   40 CONTINUE
      IF(Q.EQ.0)RETURN
      I1=0
      IJ=P2
```

```
      IL=0
      DO 55 I=1,Q
      DO 45 J=1,P
      I1=I1+1
      IJ=IJ+1
      SXY10(I1)=S10(IJ)
   45 SXY(I1)=S(IJ)
      DO 50 J=1,I
      IJ=IJ+1
      IL=IL+1
      SXX10(IL)=S10(IJ)
   50 SXX(IL)=S(IJ)
   55 CONTINUE
  100 FORMAT(2I5)
  200 FORMAT(16I5)
      RETURN
      END
      SUBROUTINE PPMSL(A,M,N,TEXT,LT,IND)
C ******* PRINT PATTERN MATRIX STORED LINEARLY
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER A
      REAL TEXT
      DIMENSION A(1),TEXT(20)
      WRITE(6,100)(TEXT(I),I=1,LT)
      DO 15 I=1,M
      IF(IND.EQ.0)GO TO 10
      L=(I*(I-1))/2
      LU=L+I
      L=L+1
      GO TO 15
   10 L=(I-1)*N
```

```
      LU=L+N
      INME=10HPPMSL
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      L=L+1
   15 WRITE(6,200)(A(J),J=L,LU)
      RETURN
  100 FORMAT(1H0,10X,8A10)
  200 FORMAT(11X,25I4)
      END
      SUBROUTINE FINOT(X,F,GG,FMT,OSC,NP,NP10,CONST)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL OSC
      INTEGER P,Q,P2,Q2,PQ
       COMMON/ORDR/P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
     +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      COMMON/PRNT/IPF
      COMMON/SIG/CYY(120),CXY(225),CXX(120),C(120),D(225),
     +CYY10(120),CXY10(225),CXX10(120),C10(120),D10(225)
      COMMON/SOL/AY(225),AX(225),B(225),G(225),R(120),
     1 V(120),TE(225),TD(225),
     +AY10(225),AX10(225),B10(225),G10(225),R10(120),V10(120),
     +TE10(225),TD10(225)
      COMMON/RES/RYY(120),RXY(225),RXX(120),
     +RYY10(120),RXY10(225),RXX10(120)
      DIMENSION X(1),GG(1),FMT(1)
      INME=10HFINOT
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
```

```
      IF(.NOT.O2)GO TO 20
      IPF=1
      LT=1
      CALL FCTGR(X,F,GG,IERR,NP,NP10)
      WRITE(6,400)
400   FORMAT(* WITHIN GROUPS RESULTS*)
      IF(.NOT.OX)GO TO 5
      CALL PMSL(M,N,D,FMT,44HREDUCED FORM MATRIX  D = ALPHA INVERSE*BETA
     1   ,LT,5,0)
      LT=0
    5 IF(.NOT.OZ)GO TO 10
      CALL PMSL(M,M,C,FMT,48HC = D*PHI*D" + ALPHA INVERSE*PSI*ALPHA INVE
     1RSE"        ,LT,5,1)
      LT=0
   10 CALL PMSL(P,P,CYY,FMT,12HSIGMA(YY)        ,LT,2,1)
      IF(.NOT.OX)GO TO 15
      CALL PMSL(Q,P,CXY,FMT,12HSIGMA(XY)       ,0,2,0)
      CALL PMSL(Q,P,CXX,FMT,12HSIGMA(XX)       ,0,2,1)
   15 WRITE(6,100)
      CALL PMSL(P,P,RYY,FMT,16HSIGMA(YY) - SYY     ,0,2,1)
      IF(.NOT.OX)GO TO 2000
      CALL PMSL(Q,P,RXY,FMT,16HSIGMA(XY) - SXY     ,0,2,0)
      CALL PMSL(Q,P,RXX,FMT,16HSIGMA(XX) - SXX     ,0,2,1)
2000  WRITE(6,401)
401   FORMAT(* BETWEEN GROUPS RESULTS*)
      IF(.NOT.OX)GO TO 500
      CALL PMSL(M10,N10,D,FMT,44HREDUCED FORM MATRIX  D = ALPHA INVERSE*
     +BETA          ,LT,5,0)
      LT=0
  500 IF(.NOT.OZ)GO TO 1000
      CALL PMSL(M10,M10,C,FMT,48HC = D*PHI*D" + ALPHA INVERSE*PSI*ALPHA
```

```
     1INVERSE"        ,LT,5,1)
      LT=0
 1000 CALL PMSL(P,P,CYY10,FMT,12HSIGMA(YY)        ,LT,2,1)
      IF(.NOT.OX)GO TO 1500
      CALL PMSL(Q,P,CXY10,FMT,12HSIGMA(XY)       ,0,2,0)
      CALL PMSL(Q,P,CXX10,FMT,12HSIGMA(XX)       ,0,2,1)
 1500 WRITE(6,100)
      CALL PMSL(P,P,RYY10,FMT,16HSIGMA(YY) - SYY     ,0,2,1)
      IF(.NOT.OX)GO TO 20
      CALL PMSL(Q,P,RXY10,FMT,16HSIGMA(XY) - SXY     ,0,2,0)
      CALL PMSL(Q,P,RXX10,FMT,16HSIGMA(XX) - SXX     ,0,2,1)
20    IF(.NOT.OO) GO TO 70
      CALL PNCH(AY,AX,B,G,R,V,TE,TD,P,Q,M,MP,NQ,MM,MN,N,M2)
      CALL PNCH(AY10,AX10,B10,G10,R10,V10,TE10,TD10,P,Q,
     +M10,MP10,NQ10,MM10,MN10,N10,M20)
70    IF(.NOT.OSC) GO TO 333
      WRITE(6,402)
402   FORMAT(* STANDARDIZED WITHIN GROUPS SOLUTION*)
      CALL SCALE(AY,AX,B,G,R,V,TE,TD,FMT,P,Q,M,N,C,D,
     +CYY,CXX)
      WRITE(6,403)
403   FORMAT(* STANDARDIZED BETWEEN GROUPS SOLUTION*)
      CALL SCALE(AY10,AX10,B10,G10,R10,V10,TE10,TD10,FMT,P,Q,
     +M10,N10,C10,D10,CYY10,CXX10)
333   CONTINUE
      NDF=(P2+Q2+PQ)*2-NOI
      IF(NDF.NE.0)GO TO 25
      WRITE(6,300)
      RETURN
   25 DF=NDF
      UK=F*2-CONST
```

```
      PLV=CHIPR(DF,UK)
      WRITE(6,200)NDF,UK,PLV
      RETURN
  100 FORMAT(1H1,5X,*RESIDUALS = SIGMA - S*)
  200 FORMAT(1H1,10X,*TEST OF GOODNESS OF FIT*//1H0,10X,*CHI SQUARE WITH
     A*,I5,* DEGREES OF FREEDOM IS*,F16.4//1H0,10X,*PROBABILITY LEVEL =*
     B,F16.4)
  300 FORMAT(1H1,10X,*TEST OF GOODNESS OF FIT*//1H0,10X,*DEGREES OF FREE
     ADOM = 0*)
      END
      SUBROUTINE SCALE(AY,AX,B,G,R,V,TE,TD,FMT,P,Q,M,N,C,D,CYY,CXX)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      DIMENSION AY(1),AX(1),G(1),R(1),V(1),TE(1),TD(1),FMT(1),
     +C(1),D(1),CYY(1),CXX(1),B(1)
      INME=10HSCALE
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      L=0
      DO 5 I=1,M
      L=L+I
       CYY(I) = SQRT(C(L))
    5 CONTINUE
      L=0
      DO 10 I=1,N
      L=L+I
      CXX(I) = SQRT(R(L))
   10 CONTINUE
      L=0
```

```
      DO 15 I=1,P
      DO 15 J=1,M
      L=L+1
      AY(L)=AY(L)*CYY(J)
   15 CONTINUE
      IF(.NOT.OX)GO TO 22
      L=0
      DO 20 I=1,Q
      DO 20 J=1,N
      L=L+1
      AX(L)=AX(L)*CXX(J)
   20 CONTINUE
   22 IF(.NOT.OE)GO TO 27
      L=0
      DO 25 I=1,M
      DO 25 J=1,M
      L=L+1
      B(L)=B(L)*CYY(J)/CYY(I)
   25 CONTINUE
   27 IF(.NOT.OX)GO TO 37
      L=0
      DO 30 I=1,M
      DO 30 J=1,N
      L=L+1
      TMP=CXX(J)/CYY(I)
      G(L)=G(L)*TMP
      D(L)=D(L)*TMP
   30 CONTINUE
      L=0
      DO 35 I=1,N
      DO 35 J=1,I
```

```
      L=L+1
      R(L)=R(L)/(CXX(I)*CXX(J))
   35 CONTINUE
   37 IF(.NOT.OZ)GO TO 42
      L=0
      DO 40 I=1,M
      DO 40 J=1,I
      L=L+1
      TMP=1.D0/(CYY(I)*CYY(J))
      V(L)=V(L)*TMP
      C(L)=C(L)*TMP
   40 CONTINUE
42    CONTINUE
      CALL SIDA(AY,AX,B,G,R,V,TE,TD,FMT,P,Q,M,N)
      IF(.NOT.OX)GO TO 45
      CALL PMSL(M,N,D,FMT,16HD (STANDARDIZED),0,4,0)
   45 IF(.NOT.OZ)GO TO 50
      CALL PMSL(M,M,C,FMT,16HC (STANDARDIZED),0,4,1)
   50 RETURN
      END
      SUBROUTINE PNCH(AY,AX,B,G,R,V,TE,TD,P,Q,M,MP,NQ,MM,MN,N2,M2)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      REAL FMT
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      DIMENSION AY(1),AX(1),B(1),G(1),R(1),V(1),TE(1),TD(1),FMT(2)
      DATA FMT/8H(5F15.7)/
      N2X=N2*N2
      WRITE(7)(AY(I),I=1,MP)
      IF(.NOT.OX)GO TO 2
```

```
      WRITE(7)(AX(I),I=1,NQ)
    2 IF(.NOT.OE)GO TO 3

      WRITE(7)(B(I),I=1,MM)
    3 IF(.NOT.OX)GO TO 5

      WRITE(7)(G(I),I=1,MN)

      WRITE(7)(R(I),I=1,N2X)
    5 IF(.NOT.OZ)GO TO 10

      WRITE(7)(V(I),I=1,MM)
10    CONTINUE
      WRITE(7)(TE(I),I=1,P)
      IF(.NOT.OX)RETURN

      WRITE(7)(TD(I),I=1,Q)
  100 FORMAT(2A10)
      RETURN
      END
      SUBROUTINE IPSMSL(M,N,A,B,C,EPS,D,E,F,DET,IERR,A1,A2,Y)
C ******* INVERT PARTITIONED SYMMETRIC MATRIX STORED LINEARLY
C                        -1
C             (A        )             (D        )
C             (         ) (INPUT)  =  (         ) (OUTPUT)
C             (B       C)             (E       F)
C
C          WHERE A,B,C,D,E,F ARE LINEARLY STORED MATRICES,AND ONLY THE
C          HALF OF SYMMETRIC MATRICES IS STORED
C             A(MXM), SYMMETRIC                  D(MXM), SYMMETRIC
C             B(NXM)                           E(NXM)
C             C(NXN), SYMMETRIC                  F(NXN), SYMMETRIC
```

```
C          EPS    IF ANY PIVOTAL ELEMENT IS LESS THAN EPS,THE MATRIX IS
C                 CONSIDERED SINGULAR AND CONTROL IS TRANSFERRED TO THE
C                 PROGRAM WITH IERR=1
C          DET     = DETERMINANT OF PARTITIONED MATRIX TO BE INVERTED
C          IERR   =0 IMPLIES THE MATRIX IS NON-SINGULAR
C          A1     INTERNAL MATRIX,MUST BE DIMENSIONED IN CALLING PROGRA
C                 LEAST N*N
C          A2     INTERNAL MATRIX, MUST BE DIMENSIONED IN CALLING PROGR
C                 LEAST THE LARGER OF N*N OR N*M
C          Y      INTERNAL VECTOR,MUST BE DIMENSIONED IN CALLING PROGRA
C                 LEAST THE LARGER OF M OR N
C        CAUTION  THIS ROUTINE CALLS SUBROUTINES ISMSL AND MMSL
C ***************************************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),C(1),D(1),E(1),F(1),A1(1),A2(1),Y(1)
      INME=10HIPSMSL
C     WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      M2=(M*(M+1))/2
      NN=N*N
      NM=N*M
      CALL ISMSL(N,C,A1,Y,DET1,EPS,IERR)
      IF(IERR.GT.0)RETURN
      CALL MMSL(N,N,M,A1,B,A2,-1,0,1)
      CALL MMSL(M,N,M,B,A2,D,1,0,0)
      DO 1 I=1,M2
      D(I)=A(I)-D(I)
    1 CONTINUE
      CALL ISMSL(M,D,D,Y,DET2,EPS,IERR)
      IF(IERR.GT.0)RETURN
```

```
      CALL MMSL(N,M,M,A2,D,E,0,-1,1)
      DO 2 I=1,NM
      E(I)=-E(I)
    2 CONTINUE
      CALL MMSL(N,M,N,B,E,A2,0,1,1)
      DO 3 I=1,NN
      A2(I)=-A2(I)
    3 CONTINUE
      K=-N
      DO 4 I=1,N
      K=K+N+1
      A2(K) = 1.0 + A2(K)
    4 CONTINUE
      CALL MMSL(N,N,N,A1,A2,F,-1,0,0)
      DET=DET1*DET2
      RETURN
      END
      SUBROUTINE ISMSL(N,A,B,Y,D,EPS,IERR)
C ***** INVERT SYMMETRIC MATRIX STORED LINEARLY
C         N    ORDER OF MATRIX
C         A    MATRIX TO BE INVERTED,STORED LINEARLY,MUST BE GRAMIAN
C         B    A INVERSE,STORED AS A VECTOR
C         Y    INTERNAL DUMMY ARRAY,MUST BE DIMENSIONED IN CALLING PROGR
C         D    DETERMINANT(A)
C         EPS  IF ANY PIVOTAL ELEMENT IS LESS THAN EPS,A IS CONSIDERED S
C              AND CONTROL IS TRANSFERRED TO THE CALLING PROGRAM WITH IE
C         IERR =0 IMPLIES A IS NON-SINGULAR
C ********************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),Y(1)
      INME=10HISMSL
C      WRITE(6,9999) INME
```

```
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      IERR=0
      NN=(N*(N+1))/2
      DO 5 I=1,NN
    5 B(I)=A(I)
      D = 1.0
      IF(N.EQ.1) GO TO 260
      DO 240 L=1,N
      F=B(1)
      IF(F.LT.EPS) GO TO 700
      D=D*F
      F = 1.0/F
      NA=1
      DO 210 I=1,N
      NA=NA+I-1
  210 Y(I)=B(NA)
      NA=0
      NB=1
      DO 220 I=2,N
      NB=NB+1
      H=Y(I)*F
      DO 220 J=2,I
      NB=NB+1
      NA=NA+1
  220 B(NA)=B(NB)-Y(J)*H
      DO 230 J=2,N
      NA=NA+1
  230 B(NA)=-Y(J)*F
  240 B(NN)=-F
      DO 250 I=1,NN
```

```
  250 B(I)=-B(I)
      RETURN
  260 L=1
      F=B(1)
      IF(F.LT.EPS) GO TO 700
      B(1) = 1.0/F
      D=F
      RETURN
  700 WRITE (6,1) L,F,D
      IERR=1
      RETURN
    1 FORMAT(1H0,10X,35HTHE MATRIX IS NOT POSITIVE DEFINITE  ,I4,2E11.4)
      END
      SUBROUTINE SIDA(AY,AX,B,G,R,V,TE,TD,FMT,P,Q,M,N)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      DIMENSION AY(1),AX(1),B(1),G(1),R(1),V(1),TE(1),TD(1),FMT(1)
      DIMENSION RNEW(120),VNEW(120)
      IF(OX) CALL MMSL(N,N,N,R,R,RNEW,0,1,0)
      IF(OZ) CALL MMSL(M,M,M,V,V,VNEW,0,1,0)
      INME=10HSIDA
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      CALL PMSL(P,M,AY,FMT,8HLAMBDA  ,0,1,0)
      IF(OX)
     1CALL PMSL(Q,N,AX,FMT,8HGAMMA   ,0,1,0)
      CALL PMSL(M,M, B,FMT,8HALPHA   ,0,1,0)
      IF(.NOT.OX)GO TO 5
      CALL PMSL(M,N,G,FMT,4HBETA,0,1,0)
```

```
      CALL PMSL(N,N, RNEW,FMT,12HSIGMA TSI   ,0,1,1)
    5 IF(OZ)
     1CALL PMSL(M,M, VNEW,FMT,12HSIGMA THETA ,0,1,1)
      CALL PMSL(1,P,TE,FMT,12HPSI EPSILON   ,0,2,0)
      IF(OX)
     1CALL PMSL(1,Q,TD,FMT,12HPSI DELTA     ,0,2,0)
      RETURN
      END
      SUBROUTINE MPSMSL(N,M,A,B,C,D,E,F,X,Y,Z,A1,A2)
C ******** MULTIPLICATION OF PARTITIONED SYMMETRIC MATRICES
C
C              (A    B")(D    E")(A    B")   (X      )
C              (       )(       )(       ) = (       )
C              (B    C )(E    F )(B    C )   (Y    Z )
C
C           WHERE A,B,C,D,E,F,X,Y,Z ARE LINEARLY STORED MATRICES, AND ON
C           THE LOWER HALF OF SYMMETRIC MATRICES IS STORED
C              A(MXM),SYMMETRIC    D(MXM),SYMMETRIC    X(MXM),SYMMETRIC
C              B(NXM)              E(NXM)              Y(NXM)
C              C(NXN),SYMMETRIC    F(NXN),SYMMETRIC    Z(NXN),SYMMETRIC
C           A1,A2    INTERNAL MATRICES STORED LINEARLY,MUST BE DIMENSION
C                    CALLING PROGRAM BY AT LEAST THE LARGER OF N*N OR M*
C           CAUTION  THIS ROUTINE CALLS SUBROUTINE MMSL
C *************************************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),C(1),D(1),E(1),F(1),X(1),Y(1),Z(1),A1(1),
     1A2(1)
      INME=10HMPSMSL
C     WRITE(6,9999) INME
 9999 FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
```

```
      MM=M*M
      M2=(M*(M+1))/2
      MN=M*N
      N2=(N*(N+1))/2
      NN=N*N
      CALL MMSL(M,M,M,A,D,A1,-1,-1,1)
      CALL MMSL(M,N,M,B,E,A2,1,0,1)
      DO 1 I=1,MM
      A1(I)=A1(I)+A2(I)
    1 CONTINUE
      CALL MMSL(M,M,M,A1,A,X,0,-1,0)
      CALL MMSL(N,M,M,B,A1,Y,0,1,1)
      CALL MMSL(M,M,N,A,E,A1,-1,1,1)
      CALL MMSL(M,N,N,B,F,A2,1,-1,1)
      DO 2 I=1,MN
      A1(I)=A1(I)+A2(I)
    2 CONTINUE
      CALL MMSL(M,N,M,A1,B,A2,0,0,0)
      DO 3 I=1,M2
      X(I)=X(I)+A2(I)
    3 CONTINUE
      CALL MMSL(N,N,M,C,A1,A2,-1,1,1)
      DO 4 I=1,MN
      Y(I)=Y(I)+A2(I)
    4 CONTINUE
      CALL MMSL(N,M,M,B,D,A1,0,-1,1)
      CALL MMSL(N,N,M,C,E,A2,-1,0,1)
      DO 5 I=1,MN
      A1(I)=A1(I)+A2(I)
    5 CONTINUE
      CALL MMSL(N,M,N,A1,B,Z,0,1,0)
```

```
      CALL MMSL(N,M,N,B,E,A1,0,1,1)
      CALL MMSL(N,N,N,C,F,A2,-1,-1,1)
      DO 6 I=1,NN
      A1(I)=A1(I)+A2(I)
    6 CONTINUE
      CALL MMSL(N,N,N,A1,C,A2,0,-1,0)
      DO 7 I=1,N2
      Z(I)=Z(I)+A2(I)
    7 CONTINUE
      RETURN
      END
      SUBROUTINE STRTVL(AY,AX,B,G,R,V,TE,TD,P,Q,M,N,MP,NQ,MM,MN,N2,M2)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      COMMON /TID/ SEC
      DIMENSION AY(1),AX(1),B(1),G(1),R(1),V(1),TE(1),TD(1)
      DIMENSION FMT(10)
      NN=N*N
      IF(SEC.EQ.99.9) GO TO 101
C *** OE=FALSE IMPLIES BETA=I
C *** OZ=FALSE IMPLIES THERE IS NO PSI
C *** OX = FASLE IMPLIES THERE ARE NO LAMBDA(X), PHI,THETA(DELTA),GAMMA
      INME=10HSTRTVL
C     WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
      READ(5,100)FMT
      READ(5,FMT)(AY(I),I=1,MP)
      IF(.NOT.OX) GO TO 5
      READ(5,100)FMT
```

```
      READ(5,FMT)(AX(I),I=1,NQ)
    5 IF(.NOT.OE) GO TO 10
      READ(5,100)FMT
      READ(5,FMT)(B(I),I=1,MM)
      GO TO 25
   10 L=0
      DO 20 I=1, M
      DO 20 J=1,M
      L=L+1
      B(L) = 0.0
      IF(I.EQ.J)B(L)=1.D0
   20 CONTINUE
   25 IF(.NOT.OX) GO TO 30
      READ(5,100)FMT
      READ(5,FMT)(G(I),I=1,MN)
      READ(5,100)FMT
      READ(5,FMT)(R(I),I=1,NN)
   30 IF(.NOT.OZ)GO TO 35
      READ(5,100)FMT
      READ(5,FMT)(V(I),I=1,MM)
   35 READ(5,100)FMT
      READ(5,FMT)(TE(I),I=1,P)
      IF(.NOT.OX) RETURN
      READ(5,100)FMT
      READ(5,FMT)(TD(I),I=1,Q)
      RETURN
  100 FORMAT(10A8)
101   READ(9)(AY(I),I=1,MP)
      IF(.NOT.OX) GO TO 105
      READ(9) (AX(I),I=1,NQ)
105   IF(.NOT.OE) GO TO 110
```

```
      READ(9) (B(I),I=1,MM)
      GO TO 125
110   L=0
      DO 200 I=1,M
      DO 200 J=1,M
      L =L+1
      B(L)=0.0
      IF(I.EQ.J) B(L)=1.0
200   CONTINUE
125   IF(.NOT.OX) GO TO 300
      READ(9)(G(I),I=1,MN)
      READ(9) (R(I),I=1,NN)
300   IF(.NOT.OZ) GO TO 350
      READ(9)(V(I),I=1,MM)
350   READ(9) (TE(I),I=1,P)
      IF(.NOT.OX) RETURN
      READ(9)(TD(I),I=1,Q)
      RETURN
      END
      SUBROUTINE PACK(MV)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      INTEGER   AY,AX,B,G,R,V,TE,TD
      INTEGER   AY10,AX10,B10,G10,R10,V10,TE10,TD10
      COMMON/SOL/AY(225),AX(225),B(225),G(225),R(120),V(120),TE(225),
     +TD(225),AY10(225),AX10(225),B10(225),G10(225),R10(120),
     +V10(120),TE10(225),TD10(225)
      COMMON/ORDR/P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
     +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      COMMON/OLP/OAY(225),OAX(225),OB(225),OG(225),OR(120),
```

```
      1OV(120),OTE(225),OTD(225),
     +OAY10(225),OAX10(225),OB10(225),OG10(225),OR10(120),
     +OV10(120),OTE10(225),OTD10(225)
      COMMON/PCKMV/IP(16)
      DIMENSION        KO(20,2),MV(1)
C *** DEFINE PATTERN MATRICES
      INME=10HPACK
C     WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
      NN=N*N
      N10N10=N10*N10
      READ(5,100)IP
      CALL PCA(IP(1),MP,AY)
      IF(OX) CALL PCA (IP(2),NQ,AX)
      CALL PCA(IP(3),MM,B)
      IF(.NOT.OX) GO TO 5
      CALL PCA (IP(4),MN,G)
      CALL PCA (IP(5),NN,R)
    5 IF(OZ) CALL PCA(IP(6),MM,V)
      CALL PCA(IP(7),P,TE)
      IF(OX) CALL PCA (IP(8),Q,TD)
      CALL PCA(IP(9),MP10,AY10)
      IF(OX) CALL PCA(IP(10),NQ10,AX10)
      CALL PCA(IP(11),MM10,B10)
      IF(.NOT.OX) GO TO 50
      CALL PCA(IP(12),MN10,G10)
      CALL PCA(IP(13),N10N10,R10)
50    IF(OZ) CALL PCA(IP(14),MM10,V10)
      CALL PCA(IP(15),P,TE10)
      CALL PCA(IP(16),Q,TD10)
```

```
      NOI=0
      NOF=0
      CALL PCB(MP,AY,NOI,NOF)
      IF(OX) CALL PCB(NQ,AX,NOI,NOF)
      CALL PCB(MM,B,NOI,NOF)
      IF(.NOT.OX) GO TO 10
      CALL PCB(MN,G,NOI,NOF)
      CALL PCB(NN,R,NOI,NOF)
   10 IF(OZ) CALL PCB(MM,V,NOI,NOF)
      CALL PCB(P,TE,NOI,NOF)
      IF(OX) CALL PCB(Q,TD,NOI,NOF)
      CALL PCB(MP10,AY10,NOI,NOF)
      IF(OX) CALL PCB(NQ10,AX10,NOI,NOF)
      CALL PCB(MM10,B10,NOI,NOF)
      IF(.NOT.OX) GO TO 1000
      CALL PCB(MN10,G10,NOI,NOF)
      CALL PCB(N10N10,R10,NOI,NOF)
 1000 IF(OZ) CALL PCB(MM10,V10,NOI,NOF)
      CALL PCB(P,TE10,NOI,NOF)
      IF(OX) CALL PCB(Q,TD10,NOI,NOF)
      IF(NOF.EQ.0) GO TO 55
   15 READ(5,200)((KO(J,L),L=1,2),J=1,16)
      IND=KO(1,1)
      IF(IND.EQ.0) GO TO 15
      L1=KO(1,2)
      GO TO (20,21,22,23,24,25,26,27,
     +2000,2100,2200,2300,2400,2500,2600,2700,55),IND
   20 MU =AY(L1)
      GO TO 30
   21 MU=AX(L1)
      GO TO 30
```

```
   22 MU=B(L1)
      GO TO 30
   23 MU=G(L1)
      GO TO 30
   24 MU=R(L1)
      GO TO 30
   25 MU=V(L1)
      GO TO 30
   26 MU=TE(L1)
      GO TO 30
   27 MU=TD(L1)
      GO TO 30
 2000 MU =AY10(L1)
      GO TO 30
 2100 MU=AX10(L1)
      GO TO 30
 2200 MU=B10(L1)
      GO TO 30
 2300 MU=G10(L1)
      GO TO 30
 2400 MU=R10(L1)
      GO TO 30
 2500 MU=V10(L1)
      GO TO 30
 2600 MU=TE10(L1)
      GO TO 30
 2700 MU=TD10(L1)
   30 DO 40 J=2,16
      IND=KO(J,1)
      IF(IND.EQ.0) GO TO 15
      L1=KO(J,2)
```

```
      GO TO (31,32,33,34,35,36,37,38,
     +3100,3200,3300,3400,3500,3600,3700,3800,55),IND
   31 AY(L1)=MU
      GO TO 40
   32 AX(L1)=MU
      GO TO 40
   33 B(L1)=MU
      GO TO 40
   34 G(L1)=MU
      GO TO 40
   35 R(L1)=MU
      GO TO 40
   36 V(L1)=MU
      GO TO 40
   37 TE(L1)=MU
      GO TO 40
   38 TD(L1)=MU
      GO TO 40
 3100 AY10(L1)=MU
      GO TO 40
 3200 AX10(L1)=MU
      GO TO 40
 3300 B10(L1)=MU
      GO TO 40
 3400 G10(L1)=MU
      GO TO 40
 3500 R10(L1)=MU
      GO TO 40
 3600 V10(L1)=MU
      GO TO 40
 3700 TE10(L1)=MU
```

```
      GO TO 40
 3800 TD10(L1)=MU
   40 CONTINUE
      GO TO 15
   55 IF(.NOT.O1) GO TO 65
      WRITE(6,300)
      WRITE(6,301)
301   FORMAT(* WITHIN GROUPS*)
      CALL PPMSL(AY,P,M,8HLAMBDA  ,1,0)
      IF(OX) CALL PPMSL(AX,Q,N,8HGAMMA   ,1,0)
      CALL PPMSL(B,M,M,8HALPHA   ,1,0)
      IF(.NOT.OX) GO TO 60
      CALL PPMSL(G,M,N,4HBETA   ,1,0)
      CALL PPMSL(R,N,N,12HSIGMA THETA ,1,0)
   60 IF(OZ) CALL PPMSL(V,M,M,12HSIGMA TSI   ,1,0)
      CALL PPMSL(TE,1,P,12HPSI EPSILON      ,2,0)
      IF(OX)  CALL PPMSL(TD,1,Q,12HPSI DELTA      ,2,0)
      CALL PPMSL(AY10,P,M10,8HLAMBDA  ,1,0)
      IF(OX) CALL PPMSL(AX10,Q,N10,8HGAMMA   ,1,0)
      CALL PPMSL(B10,M10,M10,8HALPHA   ,1,0)
      IF(.NOT.OX) GO TO 600
      CALL PPMSL(G10,M10,N10,4HBETA   ,1,0)
      CALL PPMSL(R10,N10,N10,12HSIGMA THETA ,1,0)
  600 IF(OZ) CALL PPMSL(V10,M10,M10,12HSIGMA TSI   ,1,0)
      CALL PPMSL(TE10,1,P,12HPSI EPSILON      ,2,0)
      IF(OX)  CALL PPMSL(TD10,1,Q,12HPSI DELTA      ,2,0)
   65 NOF=0
      CALL PCC(MP,AY,NOF,MV,OAY)
      IF(OX) CALL PCC(NQ,AX,NOF,MV,OAX)
      CALL PCC(MM,B,NOF,MV,OB)
      IF(.NOT.OX) GO TO 70
```

```
      CALL PCC(MN,G,NOF,MV,OG)
      CALL PCC(NN,R,NOF,MV,OR)
   70 IF(OZ) CALL PCC(MM,V,NOF,MV,OV)
      CALL PCC(P,TE,NOF,MV,OTE)
      IF(OX) CALL PCC(Q,TD,NOF,MV,OTD)
      CALL PCC(MP10,AY10,NOF,MV,OAY10)
      IF(OX) CALL PCC(NQ10,AX10,NOF,MV,OAX10)
      CALL PCC(MM10,B10,NOF,MV,OB10)
      IF(.NOT.OX) GO TO 700
      CALL PCC(MN10,G10,NOF,MV,OG10)
      CALL PCC(N10N10,R10,NOF,MV,OR10)
  700 IF(OZ) CALL PCC(MM10,V10,NOF,MV,OV10)
      CALL PCC(P,TE10,NOF,MV,OTE10)
      IF(OX) CALL PCC(Q,TD10,NOF,MV,OTD10)
      RETURN
  100 FORMAT(16I1)
  200 FORMAT(16(I2,I3))
  300 FORMAT(1H1,5X,*PARAMETER SPECIFICATIONS*)
      END
      SUBROUTINE MMSL(M,K,N,A,B,C,IA,IB,IC)
C ***** MULTIPLY MATRICES STORED LINEARLY
C        M,K,N ORDER OF MATRICES,I.E. A(MXK),B(KXN),C(MXN)
C        A,B   MATRICES TO BE MULTIPLIED,STORED AS VECTORS
C        C     AB  (A,B,C MUST BE DIMENSIONED IN CALLING PROGRAM)
C        IA,IB IA=0 , IB=0  IMPLIES  C=AB
C              IA=1 , IB=0  IMPLIES  C=A"B
C              IA=0 , IB=1  IMPLIES  C=AB"
C              IA=1 , IB=1  IMPLIES  C=A"B"
C              IA=-1  IMPLIES A IS SYMMETRIC,ONLY LOWER HALF IS STORED
C              IB=-1  IMPLIES B IS SYMMETRIC,ONLY LOWER HALF IS STORED
C        IC    IC=0   IMPLIES C IS SYMMETRIC,ONLY LOWER HALF IS STORED
C              OTHERWISE C IS A FULL MATRIX
C **************************
```

```
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),C(1)
      INME=10HMMSL
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      NN=N
      DO 17 I=1,M
      II=(I-1)*N
      IK=(I-1)*K
      I1=(I*(I-1))/2
      IF(IC.EQ.0)NN=I
      DO 17 J=1,NN
      JJ=(J-1)*K
      J1=(J*(J-1))/2
      VW = 0.0
      DO 12 L=1,K
      IF(IA)3,2,1
    1 LI=(L-1)*M+I
      GO TO 6
    2 LI=IK+L
      GO TO 6
    3 IF(I-L)5,4,4
    4 LI=I1+L
      GO TO 6
    5 LI=(L*(L-1))/2+I
    6 IF(IB)9,8,7
    7 IL=JJ+L
      GO TO 12
    8 IL=(L-1)*N+J
      GO TO 12
```

```
   9 IF(L-J)11,10,10
  10 IL=(L*(L-1))/2+J
     GO TO 12
  11 IL=J1+L
  12 VW=VW+A(LI)*B(IL)
     IF(IC)13,14,13
  13 IJ=II+J
     GO TO 17
  14 IF(I-J)15,16,16
  15 IJ=J1+I
     GO TO 17
  16 IJ=I1+J
  17 C(IJ)=VW
     RETURN
     END
     SUBROUTINE DADDY(MV,X,IND)
     LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
     INTEGER P,Q,P2,Q2,PQ
     COMMON/ORDR/ P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
    +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
     DIMENSION X(1),MV(1)
     INME=10HDADDY
C     WRITE(6,9999) INME
9999 FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
     GO TO (5,15,30),IND
C ******* UNCONSTRAINED TO CONSTRAINED
   5 DO 10 I=1,NOF
     L=NOF-I+1
     LO=MV(L)
  10 X(L)=X(LO)
```

```
      RETURN
C ******* CONSTRAINED TO UNCONSTRAINED
   15 DO 25 I=1,NOI
      DO 20 J=1,NOF
      IF(MV(J).EQ.I)GO TO 25
   20 CONTINUE
      RETURN
   25 X(I)=X(J)
      RETURN
C ******* ADD CONSTRAINED FIRST ORDER DERIVATIVES
   30 DO 40 I=1,NOI
      VW = 0.0
      DO 35 J=1,NOF
      IF(MV(J).EQ.I)VW=VW+X(J)
   35 CONTINUE
   40 X(I)=VW
      RETURN
      END
      SUBROUTINE MOVE(AY,AX,B,G,R,V,TE,TD,AY10,AX10,B10,G10,R10,
     +V10,TE10,TD10,X,IND)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      COMMON/ORDR/ P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
     +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      COMMON/OLP/OAY(225),OAX(225),OB(225),OG(225),OR(120),
     1OV(120),OTE(225),OTD(225),
     +OAY10(225),OAX10(225),OB10(225),OG10(225),OR10(120),
     +OV10(120),OTE10(225),OTD10(225)
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      DIMENSION AY(1),AX(1),B(1),G(1),R(1),V(1),TE(1),TD(1),X(1),
     +AY10(1),AX10(1),B10(1),G10(1),R10(1),V10(1),TE10(1),TD10(1)
```

```
      L=0
      INME=10HMOVE
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      N10N10=N10*N10
      NN=N*N
      CALL MVE(OAY,IND,MP,L,AY,X,1)
      IF(OX)CALL MVE(OAX,IND,NQ,L,AX,X,2)
      CALL MVE(OB,IND,MM,L,B,X,3)
      IF(.NOT.OX)GO TO 5
      CALL MVE(OG,IND,MN,L,G,X,4)
      CALL MVE(OR,IND,NN,L,R,X,5)
    5 IF(OZ)CALL MVE(OV,IND,MM,L,V,X,6)
      CALL MVE(OTE,IND,P,L,TE,X,7)
      IF(OX)CALL MVE(OTD,IND,Q,L,TD,X,8)
      CALL MVE(OAY10,IND,MP10,L,AY10,X,9)
      IF(OX) CALL MVE(OAX10,IND,NQ10,L,AX10,X,10)
      CALL MVE(OB10,IND,MM10,L,B10,X,11)
      IF(.NOT.OX) GO TO 50
      CALL MVE(OG10,IND,MN10,L,G10,X,12)
      CALL MVE(OR10,IND,N10N10,L,R10,X,13)
50    IF(OZ) CALL MVE(OV10,IND,MM10,L,V10,X,14)
      CALL MVE(OTE10,IND,P,L,TE10,X,15)
      IF(OX) CALL MVE(OTD10,IND,Q,L,TD10,X,16)
      RETURN
      END
      SUBROUTINE PCC(M,A,M1,MV,O)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL O
      INTEGER A
```

```
      DIMENSION A(1),MV(1),O(1)
      INME=10HPCC
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      DO 10 I=1,M
      IF(A(I).EQ.0) GO TO 5
      M1=M1+1
      MV(M1)=A(I)
      O(I)=.TRUE.
      GO TO 10
    5 O(I)=.FALSE.
   10 CONTINUE
      RETURN
      END
      SUBROUTINE PCB(N,A,M1,M3)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER A
      DIMENSION A(1)
      INME=10HPCB
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      DO 20 I=1,N
      IND=A(I)+1
      GO TO (20,10,15,5),IND
    5 M3=M3+1
   10 M1=M1+1
      A(I)=M1
      GO TO 20
   15 A(I)=0
```

```
   20 CONTINUE
      RETURN
      END
      SUBROUTINE PCA(IND,N,A)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER A
      DIMENSION A(1),FMT(10)
      INME=10HPCA
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      IF(IND.GE.2) GO TO 10
      DO 5 I=1,N
    5 A(I)=IND
      RETURN
   10 READ(5,100)FMT
      READ(5,FMT)(A(I),I=1,N)
      RETURN
  100 FORMAT(10A8)
      END
      SUBROUTINE IFMSL(N,B,A,PIVOT,DET)
C ***** INVERT FULL MATRIX STORED LINEARLY
C         N      ORDER OF MATRIX TO BE INVERTED
C         B      MATRIX TO BE INVERTED,STORED LINEARLY
C         A      B INVERSE, STORED LINEARLY
C         PIVOT INTERNAL DUMMY ARRAY,MUST BE DIMENSIONED IN CALLING PROG
C         DET   =0 IF B IS SINGULAR, = DETERMINANT(B) OTHERWISE
C ***************************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION A(1),B(1),PIVOT(1),IPIV(50),INDEX(50,2)
      EQUIVALENCE (IROW,JROW),(ICOL,JCOL),(AMAX,T,SWAP)
```

```
      INME=10HIFMSL
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      NN=N*N
      DO 5 J=1,NN
    5 A(J)=B(J)
      IF(N.EQ.1)GO TO 715
   10 DET = 1.0
   15 DO 20 J=1,N
   20 IPIV(J)=0
   30 DO 550 I=1,N
   40 AMAX = 0.0
   45 DO 105 J=1,N
      IRL=(J-1)*N
   50 IF(IPIV(J)-1) 60,105,60
   60 DO 100 K=1,N
      JK=IRL+K
   70 IF(IPIV(K)-1) 80,100,720
   80 IF(ABS(AMAX) - ABS(A(JK))) 85,100,100
   85 IROW=J
   90 ICOL=K
   95 AMAX=A(JK)
  100 CONTINUE
  105 CONTINUE
      IRL=(IROW-1)*N
      ICL=(ICOL-1)*N
  110 IPIV(ICOL)=IPIV(ICOL)+1
  120 IF(IPIV(ICOL)-1) 720,130,720
  130 IF(IROW-ICOL) 140,260,140
  140 DET=-DET
```

```
150 DO 200 L=1,N
    JK=IRL+L
    KJ=ICL+L
160 SWAP=A(JK)
170 A(JK)=A(KJ)
200 A(KJ)=SWAP
260 INDEX(I,1)=IROW
270 INDEX(I,2)=ICOL
    JK=ICL+ICOL
310 PIVOT(I)=A(JK)
320 DET=DET*PIVOT(I)
330 A(JK) = 1.0
340 DO 350 L=1,N
    JK=ICL+L
350 A(JK)=A(JK)/PIVOT(I)
380 DO 550 L1=1,N
390 IF(L1-ICOL) 400,550,400
400 IRL=(L1-1)*N
    JK=IRL+ICOL
    T=A(JK)
420 A(JK)=0.
430 DO 450 L=1,N
    JK=IRL+L
    KJ=ICL+L
450 A(JK)=A(JK)-A(KJ)*T
550 CONTINUE
600 DO 710 I=1,N
610 L=N+1-I
620 IF(INDEX(L,1)-INDEX(L,2)) 630,710,630
630 JROW= INDEX(L,1)
640 JCOL= INDEX(L,2)
```

```
  650 DO 705 K=1,N
      KJ=(K-1)*N
      JK=KJ+JROW
      ICL=KJ+JCOL
  660 SWAP=A(JK)
  670 A(JK)=A(ICL)
  700 A(ICL)=SWAP
  705 CONTINUE
  710 CONTINUE
      RETURN
  715 IF(B(1).EQ.0.)GO TO 720
      A(1) = 1.0/B(1)
      DET=A(1)
      RETURN
  720 WRITE (6,1)
      DET=0.
      RETURN
    1 FORMAT (23H0THE MATRIX IS SINGULAR)
      END
      FUNCTION CHIPR(DF,CHSQ)
CCHIPR     CHISQUARE PROBABILITY
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      A=.5*DF
      X=.5*CHSQ
      IF(X .GT. 0.) GO TO 100
      CHIPR=1.
      GO TO 170
  100 TERM=1.
      SUM=0.
      COFN=A
      IF(13.-X) 110,110,120
```

```
  110 IF(A-X) 140,140,120
C      CONVERGENT SERIES FOR X .LT. A OR .LT. 13.
  120 CON=1.
      FACT=-A
  130 TEMP=SUM
      SUM=SUM+TERM
      COFN=COFN+1.
      TERM=TERM*X/COFN
      IF(SUM-TEMP) 160,160,130
C      ASYMPTOTIC SERIES FOR X .GTE. A AND X .GTE. 13.
  140 CON=0.
      FACT=X
  150 TEMP=SUM
      SUM=SUM+TERM
      COFN=COFN-1.
      RATIO=COFN/X
      TERM=TERM*RATIO
      IF(SUM-TEMP) 160,160,150
  160 CHIPR = CON + EXP(ALOG(SUM) - X + A*ALOG(X) - GAML(A))/FACT
  170 RETURN
      END
      SUBROUTINE MVE(O,IND,M,L,A,X,K)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL O
      COMMON/PCKMV/IP(16)
      DIMENSION A(1),X(1),O(1)
      INME=10HMVE
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      J=K
```

```
      IF(IP(J).EQ.0)RETURN
      DO 10 I=1,M
      IF(.NOT.O(I))GO TO 10
      L=L+1
      IF(IND.EQ.1)GO TO 5
      A(I)=X(L)
      GO TO 10
    5 X(L)=A(I)
   10 CONTINUE
      RETURN
      END
      SUBROUTINE REX (INDS,IS,O1,OX,NP,FMT1,Y,Y10,SY,SY10,S,S10,OS)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
      COMMON/DISP/SYY(120), SXY(225),SXX(120),CONST,MV(80),
     +SYY10(120),SXY10(225),SXX10(120)
       COMMON/ORDR/P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
     +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      DIMENSION FMT1(1),S(1),Y(1),Y10(1),FMT(10),SY(1),S10(1)
      INME=10HREX
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      IQ=P+Q
      I2=(IQ*(IQ+1))/2
      GO TO (90,5,5,85),IS
    5 READ(5,100)FMT
      READ(5,FMT)(S(I),I=1,I2)
      READ(5,FMT)(S10(I),I=1,I2)
      GO TO (90,90,15,20),INDS
   15 IF(IS.EQ.2) GO TO 80
```

```
      READ(5,100)FMT
      READ(5,FMT)(Y(I),I=1,IQ)
      READ(5,FMT)(Y10(I),I=1,IQ)
      GO TO 75
   20 IF(IS.EQ.3) GO TO 80
      GO TO 72
72    L=0
      DO 73 I=1,IQ
      L=L+I
      Y(I)=1.0/SQRT(S(L))
73    Y10(I)=1.0/SQRT(S10(L))
   75 L=0
      DO 76 I=1,IQ
      DO 76 J=1,I
      L=L+1
      S  (L)=S  (L)*Y(I)*Y(J)
76    S10(L)=S10(L)*Y10(I)*Y10(J)
   80 CALL SELECT(S,S10,OS)
   85 IF(.NOT.O1)RETURN
      WRITE(6,601)
601   FORMAT(* WITHIN GROUPS MATRIX TO BE ANALYZED*)
      CALL PMSL(P,P,SYY,FMT1,12HSYY WITHIN  ,1,1,1)
      IF(.NOT.OX)GO TO 1000
      CALL PMSL(Q,P,SXY,FMT1,12HSXY WITHIN  ,0,1,0)
      CALL PMSL(Q,Q,SXX,FMT1,12HSXX WITHIN  ,0,1,1)
1000  WRITE (6,602)
602   FORMAT(* BETWEEN GROUPS MATRIX TO BE ANALYZED*)
      CALL PMSL(P,P,SYY10,FMT1,12HSYY BETWEEN ,1,1,1)
      IF(.NOT.OX) RETURN
      CALL PMSL(Q,P,SXY10,FMT1,12HSXY BETWEEN ,0,1,0)
      CALL PMSL(Q,Q,SXX10,FMT1,12HSXX BETWEEN ,0,1,1)
```

```
      RETURN
   90 WRITE(6,200)
      CALL EXIT
  100 FORMAT(10A8)
  200 FORMAT(1H0,*CONFLICTING INTEGER INDICATORS - CHECK YOUR PARAMETER
     ADATA CARD*)
      END
      SUBROUTINE PMSL(N,K,A,FMT,TEXT,LF,LT,IND)
C ***** PRINT MATRIX STORED LINEARLY
C         N,K   ORDER OF MATRIX,I.E. A(NXK)
C         A     MATRIX TO BE PRINTED
C         FMT   VARIABLE FORMAT WITH WHICH A IS PRINTED,SPECIFIED IN THE
C               PROGRAM THROUGH DATA CARD OR THE LIKE
C         TEXT  HOLLERITH TITLE OF MATRIX,NUMBER OF CHARACTERS IN THIS T
C               SHOULD BE A MULTIPLE OF 4
C         LF    CARRIAGE CONTROL DIGIT (I.E. LF=1 IMPLIES NEW PAGE,LF=0
C               DOUBLE SPACE  ETC)
C         LT    NUMBER OF WORDS IN TEXT,I.E. NUMBER OF CHARACTERS IN TEX
C               DIVIDED BY 4,NOT TO BE EXCEEDED BY 20  (SEE DIMENSION)
C         IND   =0 IMPLIES PRINT FULL MATRIX
C               OTHERWISE PRINT SYMMETRIC MATRIX,I.E. ONLY LOWER TRIANGU
C ********************************
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      REAL TEXT,FMT
      DIMENSION A(1),TEXT(20),FMT(1)
      INME=10HPMSL
C     WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
      LC=LF
      LO=1
```

```
      LL=1
    1 WRITE(6,11)LC,(TEXT(I),I=1,LT)
      L=MIN0(LO+9,K)
      WRITE(6,12)(I,I=LO,L)
      IF(IND.EQ.0)GO TO 2
      LL=LO
    2 DO 4 I=LL,N
      IF(IND.EQ.0)GO TO 3
      LCX=(I*(I-1))/2
      LOW=LCX+LO
      LR=LCX+MIN0(I,L)
      GO TO 4
    3 LCX=(I-1)*K
      LOW=LCX+LO
      LR=LCX+L
    4 WRITE(6,FMT)I,(A(J),J=LOW,LR)
      IF(L.EQ.K)RETURN
      LO=LO+10
      LC=0
      GO TO 1
   11 FORMAT(1H0,I1,10X,8A10)
   12 FORMAT(1H0,10X,10I11)
      END
      SUBROUTINE STEDE (N,X,Y,G,H,D,F,O3,IND,NP,NP10)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL O
      LOGICAL LAV
      COMMON /FSP/ TRIAL,FACT,AA,BB,EPS,MAXTRY,ITER,O
      DIMENSION X(1),Y(1),G(1),H(1),D(1)
      INME=10HSTEDE
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
```

```
      MAXTRY=15
      PR = .05
      TRIAL = .1
      FACT = .0001
      AA = 2.0
      BB = .75
      EPS = .00005
      O=.FALSE.
      IF(.NOT.O3)O=.TRUE.
      CALL FCTGR  (X,F0,G,IND,NP,NP10)
      IF(IND.GT.0) RETURN
      IF( O ) WRITE (6,11)
      ITER=0
      NOCH=0
  200 ITER=ITER+1
      IF(LAV(N,G,EPS)) RETURN
      IF(NOCH.GE.2) GO TO 230
      DO 210 I=1,N
  210 D(I)=-G(I)
      CALL SEARCH (N,X,Y,G,H,D,F0,F,IND,NP,NP10)
      IF(IND.GT.2.OR.F.GT.F0) RETURN
      W=(F0-F)/F0
      F0=F
      DO 220 I=1,N
      X(I)=Y(I)
  220 G(I)=H(I)
      NOCH=NOCH+1
      IF(W.GT.PR) NOCH=0
      GO TO 200
  230 IND=4
      RETURN
```

```
   11 FORMAT (1H1,10X,42HBEHAVIOR UNDER STEEPEST DESCENT ITERATIONS ////
     1             11X,4HITER,2X,3HTRY,4X,8HABSCISSA,10X,5HSLOPE,13X,
     18HFUNCTION )
      END
      SUBROUTINE SEARCH(N,X,Y,G,H,D,F0,F,IND,NP,NP10)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL O
      COMMON /FSP/ TRIAL,FACT,AA,BB,EPS,MAXTRY,ITER,O
      DIMENSION X(1),Y(1),G(1),H(1),D(1),P0(3),P1(3),P2(3),P3(3),P4(3)
      INME=10HSEARCH
C     WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
      IDF=0
      IND=0
      SUM=0.
      DO 210 I=1,N
  210 SUM=SUM+D(I)**2
      SUM = 1./SQRT(SUM)
      DO 220 I=1,N
  220 D(I)=SUM*D(I)
      SUM=0.
      DO 230 I=1,N
  230 SUM=SUM+G(I)*D(I)
      IF(SUM.LT.0.) GO TO 240
      IND=3
      RETURN
  240 P0(1)=0.
      P0(2)=SUM
      P0(3)=F0
      DO 250 I=1,3
```

```
    P2(I)=P0(I)
250 P1(I)=P0(I)
    IPS=0
    P4(1)=TRIAL
    ICOUNT=0
    IF( O ) WRITE (6,11) ITER,ICOUNT,(P0(I),I=1,3)
    DO 310 ICOUNT=1,MAXTRY
    DO 260 I=1,N
260 Y(I)=X(I)+P4(1)*D(I)
    CALL FCTGR  (Y,F,H,IND,NP,NP10)
    CALL TIRP(IND)
    IF(IND-1) 265,280,320
265 SX=P4(1)
    IDF=1
    P4(3)=F
    SUM=0.
    DO 270 I=1,N
270 SUM=SUM+H(I)*D(I)
    P4(2)=SUM
    IF( O ) WRITE (6,12) ICOUNT,(P4(I),I=1,3)
    CALL POINT (P0,P1,P2,P3,P4,IPS)
    IF(IPS.EQ.2) RETURN
    GO TO 310
280 IF( O ) WRITE (6,12) ICOUNT,P4(1)
    IF(IDF.GT.0) GO TO 290
    P4(1)=0.5*P4(1)
    GO TO 310
290 DO 300 I=1,N
300 Y(I)=X(I)+SX*D(I)
    CALL FCTGR  (Y,F,H,IND,NP,NP10)
    RETURN
```

```
  310 CONTINUE
      IND=2
  320 RETURN
   11 FORMAT(1H0,9X,2I5,3(3X,E15.8))
   12 FORMAT(15X,I5,3(3X,E15.8))
      END
      SUBROUTINE FLEPOW (N,X,Y,G,H,D,E,F,O3,IND,OA,NP,NP10)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL O
      LOGICAL LAV
      COMMON /FSP/ TRIAL,FACT,AA,BB,EPS,MAXTRY,ITER,O
      DIMENSION X(1),Y(1),G(1),H(1),D(1),E(1)
      INME=10HFLEPOW
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      MAXITE=100
      MAXTRY=20
      TRIAL = .1
      FACT = .000001
      AA = 2.0
      BB = .75
      EPS = .00005
      O=.FALSE.
      IF(.NOT.O3)O=.TRUE.
      CALL FCTGR  (X,F0,G,IND,NP,NP10)
      IF(IND.GT.0) RETURN
      IF( O ) WRITE (6,11)
      DO 300 ITER=1,MAXITE
      IF(LAV(N,G,EPS)) RETURN
      CALL SYMAMU (N,E,G,D)
```

```
      CALL SEARCH (N,X,Y,G,H,D,F0,F,IND,NP,NP10)
      IF(OA .AND.(((F0-F)/F).LT.0.05))RETURN
      IF(IND.GT.0) GO TO 240
      F0=F
      DO 210 I=1,N
      C1=Y(I)
      C2=H(I)
      Y(I)=TRIAL*D(I)
      H(I)=H(I)-G(I)
      X(I)=C1
  210 G(I)=C2
      CALL SYMAMU(N,E,H,D)
      C1 = C2 = 0.
      DO 220 I=1,N
      C1=C1+H(I)*Y(I)
  220 C2=C2-H(I)*D(I)
      C1 = 1.0/C1
      C2 = 1.0/C2
      L=0
      DO 230 I=1,N
      V1=C1*Y(I)
      V2=C2*D(I)
      DO 230 J=1,I
      L=L+1
  230 E(L)=E(L)+V1*Y(J)-V2*D(J)
      GO TO 300
  240 IF(IND.GT.2) RETURN
      IF(F.GT.F0) GO TO 260
      F0=F
      DO 250 I=1,N
      X(I)=Y(I)
```

```
  250 G(I)=H(I)
      GO TO 300
  260 DO 270 I=1,N
  270 D(I)=-G(I)
      CALL SEARCH(N,X,Y,G,H,D,F0,F,IND,NP,NP10)
      IF(IND.GT.0) RETURN
      F0=F
      DO 280 I=1,N
      X(I)=Y(I)
  280 G(I)=H(I)
  300 CONTINUE
      IND=4
      RETURN
   11 FORMAT (1H1,10X,32HBEHAVIOR UNDER FLEPOW ITERATIONS ////
     1           11X,4HITER,2X,3HTRY,4X,8HABSCISSA,10X,5HSLOPE,13X,
     18HFUNCTION )
      END
      SUBROUTINE INTPOL (P1,P2,P3)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION P1(3),P2(3),P3(3)
      INME=10HINTPOL
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      Z=(3./(P2(1)-P1(1)))*(P1(3)-P2(3))+P1(2)+P2(2)
      TEMP = P1(2) + P2(2) + 2.*Z
      TEMP = ABS(TEMP)
      IF(TEMP.LT..0000005)GO TO 200
      W=Z**2-P1(2)*P2(2)
      IF(W.LE.0.0)GO TO 200
      W = SQRT(W)
      P3(1)=P1(1)+(1.-((P2(2)+W-Z)/(P2(2)-P1(2)+2.*W)))*(P2(1)-P1(1))
```

```
      RETURN
  200 P3(1)=P1(1)-P1(2)*((P2(1)-P1(1))/(P2(2)-P1(2)))
      RETURN
      END
      SUBROUTINE POINT (P0,P1,P2,P3,P4,IPS)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      LOGICAL B1,B2,B3,B4,B5
      COMMON /FSP/ TRIAL,FACT,AA,BB,EPS,MAXTRY,ITER,O
      DIMENSION P0(3),P1(3),P2(3),P3(3),P4(3)
      INME=10HPOINT
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      B1 = ABS(P4(2)).LT.FACT*ABS(P0(2))
      B2=P4(3).GT.P0(3)
      B3=P4(2).GT.0.
      B4=(B1.OR..NOT.B3).AND.B2
      B5=P4(2).GT.P2(2).AND.P2(2).GE.P1(2)
      IF(.NOT.B4) GO TO 220
      IPS=0
      P4(1)=BB*P4(1)
      DO 210 I=1,3
      P1(I)=P0(I)
  210 P2(I)=P0(I)
      RETURN
  220 IF(.NOT.B1) GO TO 230
      IPS=2
      TRIAL=P4(1)
      RETURN
  230 IF(.NOT.B3) GO TO 250
      IPS=1
```

```
      DO 240 I=1,3
  240 P3(I)=P4(I)
      CALL INTPOL(P2,P3,P4)
      RETURN
  250 DO 260 I=1,3
      P1(I)=P2(I)
  260 P2(I)=P4(I)
      IF(IPS.NE.1) GO TO 270
      CALL INTPOL(P2,P3,P4)
      RETURN
  270 IF(.NOT.B5) GO TO 280
      CALL INTPOL(P1,P2,P4)
      RETURN
  280 P4(1)=AA*P4(1)
      RETURN
      END
      LOGICAL FUNCTION LAV(N,G,EPS)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION G(1)
      LAV=.FALSE.
      DO 200 I=1,N
      IF(ABS(G(I)).GT.EPS)RETURN
  200 CONTINUE
      LAV=.TRUE.
      RETURN
      END
      SUBROUTINE TIRP (IND)
      COMMON /TID/ SEC
      T = SECOND(VAD)
      INME=10HTIRP
C     WRITE(6,9999) INME
 9999 FORMAT(1X,A10)
C     CALL PDUMP(66507B,66510B,4)
```

```
      IF(VAD.GT.SEC) IND=5
      RETURN
      END
      SUBROUTINE SYMAMU(N,E,G,D)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      DIMENSION E(1),G(1),D(1)
      INME=10HSYMAMU
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
      DO 15 I=1,N
      SUM = 0.0
      IT=(I*(I-1))/2
      DO 10 J=1,N
      L=IT+J
      IF(I.GE.J)GO TO 10
      L=(J*(J-1))/2+I
   10 SUM=SUM-E(L)*G(J)
   15 D(I)=SUM
      RETURN
      END
      FUNCTION GAML(A)
C
C     ..................................................................
C
C        FUNCTION GAML
C
C        PURPOSE
C           COMPUTES THE NATURAL LOGARITHM OF THE GAMMA FUNCTION
C           OF A NUMBER.
C
C        USAGE
```

```
C           G = GAML(A)
C
C        DESCRIPTION OF PARAMETERS
C           A - - - INPUT
C           G - - - NATURAL LOGARITHM OF THE GAMMA FUNCTION OF A
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C     ..................................................................
C
      W=A
      TEMP=0.
      IF(W-13.) 140,140,150
  140 N=14.-W
      TEMP=1.
      DO 145 I=1,N
      TEMP=TEMP*W
  145 W=W+1.
      TEMP=ALOG(TEMP)
  150 W2=W*W
      GAML=(.0833333333-(.00277777777-.000793650793/W2)/W2)/W+.918938533
     2  -W+(W-.5)*ALOG(W)-TEMP
      RETURN
      END
      SUBROUTINE FCTGR(X,F,GG,IERR,NP,NP10)
      LOGICAL O1,O2,O3,OO,OX,OZ,OE,OA
      INTEGER P,Q,P2,Q2,PQ
```

```
      COMMON/ORDR/ P,Q,M,N,P2,Q2,PQ,MP,NQ,M2,N2,MM,MN,NOI,NOF,
     +M10,N10,MP10,N20,MM10,NQ10,M20,MN10
      COMMON/COND/IS,INDS,OX,OZ,OE,OA,OO,O1,O2,O3
      COMMON/SOL/AY(225),AX(225),B(225),G(225),R(120),
     1 V(120),TE(225),TD(225),
     +AY10(225),AX10(225),B10(225),G10(225),R10(120),V10(120),
     +TE10(225),TD10(225)
      COMMON/DISP/SYY(120), SXY(225),SXX(120),CONST,MV(80),
     +SYY10(120),SXY10(225),SXX10(120)
      COMMON/RES/RYY(120),RXY(225),RXX(120),
     + RYY10(120),RXY10(225),RXX10(120)
      COMMON/SIG/CYY(120),CXY(225),CXX(120),C(120),D(225),
     +CYY10(120),CXY10(225),CXX10(120),C10(120),D10(225)
      COMMON/PRNT/IPF
      DIMENSION X(1),GG(1),PIVOT(15),              A1(225),A2(225),
     1A(225),A3(225),A4(120),A5(225),A6(225),A7(120),A8(225),A9(225)
      DIMENSION A0(225),A10(225),A20(225),A30(225),A40(120),A50(225),
     +A60(225),A70(120),A80(225),A90(225)
      DIMENSION EXYY(120),EXXY(225),EXXX(120)
      DIMENSION RNEW(120),RNEW10(120),VNEW(120),VNEW10(120),A400(120),
     +A4000(120)
      EQUIVALENCE(RYY,A5),(RXY,A6),(RXX,A7)
      EQUIVALENCE (RYY10,A50),(RXY10,A60),(RXX10,A70)
C      WRITE(6,5550) NP,NP10,NOI,NOF
5551  FORMAT(2(1X,F6.3))
5550  FORMAT(2(1X,I6))
555   FORMAT(* A*)
      INME=10HFCTGR
C      WRITE(6,9999) INME
9999  FORMAT(1X,A10)
C      CALL PDUMP(66507B,66510B,4)
556   FORMAT(* B*)
```

```
557   FORMAT(* C*)
558   FORMAT(* D*)
559   FORMAT(* E*)
560   FORMAT(* F*)
561   FORMAT(* G*)
562   FORMAT(* H*)
563   FORMAT(* I*)
564   FORMAT(* J*)
565   FORMAT(* K*)
566   FORMAT(* L*)
C      WRITE(6,555)
      IERR=0
      NPNP10=NP10*NP
      NPNP9=NP10*(1-NP)
      IF(IPF.EQ.1)GO TO 1
      IF(NOI.LT.NOF)CALL DADDY(MV,X,1)
      CALL MOVE(AY,AX,B,G,R,V,TE,TD,AY10,AX10,B10,G10,R10,V10,TE10,TD10,
     +X,0)
      IF(NOI.LT.NOF)CALL DADDY(MV,X,2)
1     IF(.NOT.OX) GO TO 777
C RNEW = R * R?
      CALL MMSL(N,N,N,R,R,RNEW,0,1,0)
C RNEW10 = R10 * R10/
      CALL MMSL(N10,N10,N10,R10,R10,RNEW10,0,1,0)
777   IF(.NOT.OZ) GO TO 778
C VNEW = V * V/
      CALL MMSL(M,M,M,V,V,VNEW,0,1,0)
C VNEW10 = V10 * V10/
      CALL MMSL(M10,M10,M10,V10,V10,VNEW10,0,1,0)
778   CONTINUE
C ******* COMPUTE SIGMA
C      WRITE(6,556)
```

```
      IF(OZ.OR.OX)GO TO 10
      DO 5 I=1,P2
      CYY(I) = 0.0
      CYY10(I)=0.0
    5 CONTINUE
      GO TO 80
   10 IF(.NOT.OE)GO TO 15
C      WRITE(6,557)
      CALL IFMSL(M,B,A,PIVOT,DETB)
      CALL IFMSL(M10,B10,A0,PIVOT,DETB10)
      IF((DETB.NE.0.0).AND.(DETB10.NE.0.0)) GO TO 15
      IERR=1
C      WRITE(6,1000)
      RETURN
   15 IF(.NOT.OZ)GO TO 35
C      WRITE(6,558)
      IF(OE)GO TO 25
      DO 20 I=1,M2
      C(I)=VNEW(I)
   20 CONTINUE
      DO 10020 I=1,M20
      C10(I)=VNEW10(I)
10020 CONTINUE
      GO TO 30
   25 CALL MMSL(M,M,M,A,VNEW,D,0,-1,1)
      CALL MMSL(M,M,M,D,A,C,0, 1,0)
      CALL MMSL(M10,M10,M10,A0,VNEW10,D10,0,-1,1)
      CALL MMSL(M10,M10,M10,D10,A0,C10,0,1,0)
C      WRITE(6,559)
   30 IF(.NOT.OX)GO TO 75
   35 IF(OE)GO TO 45
```

```
      DO 40 I=1,MN
      D(I)=G(I)
   40 CONTINUE
      DO 10040 I=1,MN10
      D10(I)=G10(I)
10040 CONTINUE
      GO TO 50
   45 CALL MMSL(M,M,N,A,G,D,0,0,1)
      CALL MMSL(M10,M10,N10,A0,G10,D10,0,0,1)
   50 CALL MMSL(M,N,N,D,RNEW,CXY,0,-1,1)
      CALL MMSL(M,N,M,CXY,D,CXX,0,1,0)
      CALL MMSL(M10,N10,N10,D10,RNEW10,CXY10,0,-1,1)
      CALL MMSL(M10,N10,M10,CXY10,D10,CXX10,0,1,0)
C      WRITE(6,560)
      IF(OZ)GO TO 60
      DO 55 I=1,M2
      C(I)=CXX(I)
   55 CONTINUE
      DO 10055 I=1,M20
      C10(I)=CXX10(I)
10055 CONTINUE
      GO TO 70
   60 DO 65 I=1,M2
      C(I)=C(I)+CXX(I)
   65 CONTINUE
      DO 10065 I=1,M20
      C10(I)=C10(I)+CXX10(I)
10065 CONTINUE
C      WRITE(6,561)
   70 CALL MMSL(Q,N,M,AX,CXY,A2,0,1,1)
      CALL MMSL(Q,N,N,AX,RNEW,CXY,0,-1,1)
```

```
      CALL MMSL(Q,N,Q,CXY,AX,CXX,0,1,0)
      CALL MMSL(Q,N10,M10,AX10,CXY10,A20,0,1,1)
      CALL MMSL(Q,N10,N10,AX10,RNEW10,CXY10,0,-1,1)
      CALL MMSL(Q,N10,Q,CXY10,AX10,CXX10,0,1,0)
      L=0
      DO 72 I=1,Q
      L=L+I
      CXX(L)=CXX(L)+TD(I)**2
      CXX10(L)=CXX10(L)+TD10(I)**2
   72 CONTINUE
      CALL MMSL(Q,M,P,A2,AY,CXY,0,1,1)
      CALL MMSL(Q,M10,P,A20,AY10,CXY10,0,1,1)
   75 CALL MMSL(P,M,M,AY,C,A1,0,-1,1)
      CALL MMSL(P,M,P,A1,AY,CYY,0,1,0)
      CALL MMSL(P,M10,M10,AY10,C10,A10,0,-1,1)
      CALL MMSL(P,M10,P,A10,AY10,CYY10,0,1,0)
   80 L=0
      DO 85 I=1,P
      L=L+I
      CYY(L)=CYY(L)+TE(I)**2
      CYY10(L)=CYY10(L)+TE10(I)**2
   85 CONTINUE
90    DO 10000 I=1,P2
10000 EXYY(I)=CYY(I)+NP*CYY10(I)
C      WRITE(6,5553)(EXYY(I),CYY(I),CYY10(I),I=1,20)
5553  FORMAT(3(1X,F9.2))
      IF(.NOT.OX) GO TO 9000
      DO 10001 I=1,PQ
10001 EXXY(I)=CXY(I)+NP*CXY10(I)
      DO 10002 I=1,Q2
10002 EXXX(I)=CXX(I)+NP*CXX10(I)
```

```
      GO TO 95
C ******* COMPUTE TRACE(S*SIGMA INVERSE)
C      WRITE(6,557)
9000  IF(IPF.EQ.1)GO TO 125
      CALL ISMSL(P,CYY,C,PIVOT,DET,5.E-12,IERR)
      CALL ISMSL(P,EXYY,C10,PIVOT,DET10,5.E-12,IERR)
C      WRITE(6,5554) (C(I),C10(I),I=1,20)
      GO TO 96
   95 CALL IPSMSL(P,Q,CYY,CXY,CXX,5.E-12,C,A3,A4,DET,IERR,A5,A6,PIVOT)
      CALL IPSMSL(P,Q,EXYY,EXXY,EXXX,5.E-12,C10,A30,A40,DET10,
     +IERR10,A50,A60,PIVOT)
96    IF(IERR.EQ.0.AND.IERR10.EQ.0) GO TO 97
C      WRITE(6,2000)
      IF(IERR.EQ.0) IERR=IERR10
      RETURN
   97 TR = 0.0
      TR10=0.0
      DO 100 I=1,P2
      TR=TR+SYY(I)*C(I)
      TR10=TR10+SYY10(I)*C10(I)
  100 CONTINUE
      TR = 2.0 * TR
      TR10=2.0*TR10
      L=0
      DO 105 I=1,P
      L=L+I
      TR=TR-SYY(L)*C(L)
      TR10=TR10-SYY10(L)*C10(L)
  105 CONTINUE
      IF(.NOT.OX)GO TO 122
      VW = 0.0
```

```
      VW10=0.0
      DO 110 I=1,Q2
      VW=VW+SXX(I)*A4(I)
      VW10=VW10+SXX10(I)*A40(I)
  110 CONTINUE
      VW = 2.0 * VW
      VW10=VW10*2.0
      L=0
      DO 115 I=1,Q
      L=L+I
      VW=VW-SXX(L)*A4(L)
      VW10=VW10-SXX10(L)*A40(L)
  115 CONTINUE
      TR=TR+VW
      TR10=TR10+VW10
      VW = 0.0
      VW10=0.0
      DO 120 I=1,PQ
      VW=VW+SXY(I)*A3(I)
      VW10=VW10+SXY10(I)*A30(I)
  120 CONTINUE
      TR = TR + 2.0 * VW
      TR10=TR10+2.0*VW10
C ******* COMPUTE F(X)
122   F=((NP-1)*ALOG(DET)+ALOG(DET10)+NP*TR+TR10)*0.5*NP10
C      WRITE(6,5551) TR,TR10
C      WRITE(6,5555) DET,DET10
5555  FORMAT(2(1X,F10.2))
C ******* COMPUTE RESIDUALS
C      WRITE(6,558)
  125 DO 130 I=1,P2
```

```
      RYY(I)=CYY(I)-SYY(I)
      RYY10(I)=EXYY(I)-SYY10(I)
  130 CONTINUE
      IF(.NOT.OX)GO TO 145
      DO 135 I=1,PQ
      RXY(I)=CXY(I)-SXY(I)
      RXY10(I)=EXXY(I)-SXY10(I)
  135 CONTINUE
      DO 140 I=1,Q2
      RXX(I)=CXX(I)-SXX(I)
      RXX10(I)=EXXX(I)-SXX10(I)
  140 CONTINUE
  145 IF(IPF.EQ.1)RETURN
C ******* COMPUTE OMEGA
      IF(OX) GO TO 150
      CALL MMSL(P,P,P,C,SYY,A8,-1,-1,1)
      CALL MMSL(P,P,P,A8,C,CYY,0,-1,0)
      CALL MMSL(P,P,P,C10,SYY10,A80,-1,-1,1)
      CALL MMSL(P,P,P,A80,C10,CYY10,0,-1,0)
C     WRITE(6,5554) (CYY(I),CYY10(I),I=1,20)
      DO 11000 I=1,P2
      CYY(I)=-(NPNP9*C(I)-NP10*C10(I)+NPNP10*CYY(I)+NP10*CYY10(I))
      CYY10(I)=-NP*NP10*(CYY10(I)-C10(I))
11000 CONTINUE
C     WRITE(6,5554) (CYY(I),CYY10(I),I=1,20)
5554  FORMAT(2(1X,F8.3))
      IF(OZ) GO TO 155
      GO TO 192
150   CALL MPSMSL(Q,P,C,A3,A4,SYY,SXY,SXX,CYY,CXY,CXX,A8,A9)
      CALL MPSMSL(Q,P,C10,A30,A40,SYY10,SXY10,SXX10,CYY10,CXY10,
     +CXX10,A80,A90)
```

```
      DO 11001 I=1,P2
      CYY(I)=-(NPNP9*C(I)-NP10*C10(I)+NPNP10*CYY(I)+NP10*CYY10(I))
      CYY10(I)=-NP*NP10*(CYY10(I)-C10(I))
11001 CONTINUE
      DO 11002 I=1,PQ
      CXY(I)=-(NPNP9*A3(I)-NP10*A30(I)+NPNP10*CXY(I)+NP10*CXY10(I))
      CXY10(I)=-NP*NP10*(CXY10(I)-A30(I))
11002 CONTINUE
      DO 11003 I=1,Q2
      CXX(I)=-(NPNP9*A4(I)-NP10*A40(I)+NPNP10*CXX(I)+NP10*CXX10(I))
      CXX10(I)=-NP*NP10*(CXX10(I)-A40(I))
11003 CONTINUE
C ******* COMPUTE DERIVATIVES
C ********************** COMPUTE DF/DLAMBDA Y
  155 CALL MMSL(P,P,M,CYY,A1,A3,-1,0,1)
      CALL MMSL(P,P,M10,CYY10,A10,A30,-1,0,1)
C      WRITE(6,5554) (A3(I),A30(I),I=1,20)
      IF(.NOT.OX)GO TO 166
      CALL MMSL(P,Q,M,CXY,A2,A6,1,0,1)
      CALL MMSL(P,Q,M10,CXY10,A20,A60,1,0,1)
      DO 160 I=1,MP
      A3(I)=A3(I)+A6(I)
  160 CONTINUE
      DO 1160 I=1,MP10
      A30(I)=A30(I)+A60(I)
1160  CONTINUE
C *********************** COMPUTE DF/DLAMDA X
      CALL MMSL(P,M,N,AY,D,A5,0,0,1)
      CALL MMSL(Q,P,N,CXY,A5,A2,0,0,1)
      CALL MMSL(Q,Q,N,CXX,AX,A1,-1,0,1)
      DO 165 I=1,NQ
```

```
      A2(I)=A2(I)+A1(I)
  165 CONTINUE
      CALL MMSL(Q,N,N,A2,RNEW,A1,0,-1,1)
      CALL MMSL(P,M10,N10,AY10,D10,A50,0,0,1)
      CALL MMSL(Q,P,N10,CXY10,A50,A20,0,0,1)
      CALL MMSL(Q,Q,N10,CXX10,AX10,A10,-1,0,1)
      DO 1165 I=1,NQ10
      A20(I)=A20(I)+A10(I)
 1165 CONTINUE
      CALL MMSL(Q,N10,N10,A20,RNEW10,A10,0,-1,1)
C *********************** COMPUTE DF/DBETA
  166 IF(OE)GO TO 168
      DO 167 I=1,MP
  167 A6(I)=AY(I)
      DO 1167 I=1,MP10
 1167 A60(I)=AY10(I)
      GO TO 169
  168 CALL MMSL(P,M,M,AY,A,A6,0,0,1)
      CALL MMSL(P,M10,M10,AY10,A0,A60,0,0,1)
      IF(.NOT.OE)GO TO 171
  169 CALL MMSL(M,P,M,A6,A3,A8,1,0,1)
      DO 170 I=1,MM
      A8(I)=-A8(I)
  170 CONTINUE
      CALL MMSL(M10,P,M10,A60,A30,A80,1,0,1)
      DO 1170 I=1,MM10
      A80(I)=-A80(I)
 1170 CONTINUE
C      WRITE(6,5554) (A8(I),A80(I),I=1,20)
  171 IF(.NOT.OX)GO TO 186
C *********************** COMPUTE DF/DGAMMA
```

```
      CALL MMSL(P,P,N,CYY,A5,A9,-1,0,1)
      CALL MMSL(P,Q,N,CXY,AX,D,1,0,1)
      K=P*N
      DO 175 I=1,K
      D(I)=D(I)+A9(I)
  175 CONTINUE
      CALL MMSL(M,P,N,A6,D,A9,1,0,1)
      CALL MMSL(M,N,N,A9,RNEW,A,0,-1,1)
      CALL MMSL(P,P,N10,CYY10,A50,A90,-1,0,1)
      CALL MMSL(P,Q,N10,CXY10,AX10,D10,1,0,1)
      K=P*N10
      DO 1175 I=1,K
      D10(I)=D10(I)+A90(I)
 1175 CONTINUE
      CALL MMSL(M10,P,N10,A60,D10,A90,1,0,1)
      CALL MMSL(M10,N10,N10,A90,RNEW10,A0,0,-1,1)
C ********************** COMPUTE DF/DPHI
      CALL MMSL(N,P,N,A5,D,A9,1,0,0)
      CALL MMSL(N,Q,N,AX,A2,A5,1,0,0)
      L=0
      DO 185 I=1,N
      DO 180 J=1,I
      L=L+1
      A4(L)=A9(L)+A5(L)
  180 CONTINUE
  185 CONTINUE
      CALL MMSL(N,N,N,A4,R,A400,-1,0,1)
      CALL MMSL(N10,P,N10,A50,D10,A90,1,0,0)
      CALL MMSL(N10,Q,N10,AX10,A20,A50,1,0,0)
      L=0
```

```
      DO 1185 I=1,N10
      DO 1180 J=1,I
      L=L+1
      A40(L)=A90(L)+A50(L)
1180  CONTINUE
1185  CONTINUE
      CALL MMSL(N10,N10,N10,A40,R10,A4000,-1,0,1)
C ********************** COMPUTE DF/DPSI
186   IF(.NOT.OZ)GO TO 192
      CALL MMSL(M,P,P,A6,CYY,D,1,-1,1)
      CALL MMSL(M,P,M,D,A6,A7,0,0,0)
      CALL MMSL(M,M,M,A7,V,A4,-1,0,1)
      CALL MMSL(M10,P,P,A60,CYY10,D10,1,-1,1)
      CALL MMSL(M10,P,M10,D10,A60,A70,0,0,0)
      CALL MMSL(M10,M10,M10,A70,V10,A40,-1,0,1)
C      WRITE(6,5554)(A7(I),A70(I),I=1,20)
C ********************** COMPUTE DF/DTHETA EPS  AND  DF/DTHETA DELTA
  192 L=0
      DO 195 I=1,P
      L=L+I
      A5(I)=CYY(L)*TE(I)
      A50(I)=CYY10(L)*TE10(I)
  195 CONTINUE
C      WRITE(6,5554)(A5(I),A50(I),I=1,20)
      IF(.NOT.OX)GO TO 205
      L=0
      DO 200 I=1,Q
      L=L+I
      A9(I)=CXX(L)*TD(I)
      A90(I)=CXX10(L)*TD10(I)
  200 CONTINUE
```

```
 205 CALL MOVE(A3,A1,A8,A,A400,A4,A5,A9,A30,A10,A80,A0,A4000,A40,A50,
    +A90,GG,1)
C     WRITE(6,5554) (X(I),GG(I),I=1,80)
     IF(NOI.LT.NOF)CALL DADDY(MV,GG,3)
     RETURN
1000 FORMAT(*0BETA IS SINGULAR*)
2000 FORMAT(*0SIGMA IS NOT POSITIVE DEFINITE*)
     END
```