SOME NUMERICAL COMPUTATIONS IN LINEAR ESTIMATION

by

Binay K. Bhattacharya

School of Computer Science
McGill University

January, 1978

A thesis submitted to the Faculty of Graduate Studies and Research, in partial fulfillment of the requirements for the degree of Master of Science.

© Binay K. Bhattacharya 1978

Abstract

In this thesis we have considered various types of generalized linear least squares problems. These can be treated by minimizing a sum of squares subject to linear equality constraints. Methods for solving these problems which are available in the literature can often be shown to be numerically unstable or computationally inefficient. The main effort of this thesis has been directed towards developing reliable numerically stable algorithms for certain such problems. Since we also want the most efficient numerically stable algorithms, we have made careful use of the structure of any particular system.

In this thesis we have presented a numerically stable algorithm for solving generalized linear least squares problems which is based on the work carried out by Paige [18]. We have developed a very efficient numerically stable algorithm for obtaining the minimum 2-norm solution of a structured underdetermined system. We have then considered three parameter estimation problems which can be formulated as generalized least squares problems. They are: a repeatable experiment with a general linear model, grouping of equations, and estimation in a dynamical system. We have presented numerically stable algorithms for solving such problems and a comparison has been made with the existing numerically unstable methods. These algorithms have been developed jointly with C. Paige following on the original work of Paige [18].

RÉSUMÉ

Dans cette thèse nous avons considéré une variété de types de problèmes moindres carrés linéaires généralisés. Ceux-ci peuvent être traités en minimisant une somme de carrés sujette à des contraintes d'égalité linéaire. On peut souvent démontrer que les méthodes de résolution de ces problèmes disponibles dans la littérature sont instables numériquement ou inefficaces quant aux calculs à effectuer. L'effort principal de cette thèse est dirigé vers le développement d'algorithmes fiables et numériquement stables pour certains problèmes de ce genre. Puisque nous désirons des algorithmes qui soient aussi des plus efficaces, nous avons utilisé avec précautions la structure propre de chaque système considéré.

Dans cette thèse nous avons présenté un algorithme rapide et numériquement stable pour la résolution de problèmes moindres carrés linéaires généralisés, basé sur le travail entrepris par Paige (18). Nous avons développé un algorithme numériquement stable et très efficace permettant d'obtenir la solution 2-norme minimale d'un système structuré indéterminé. Nous avons ensuite considéré trois problèmes d'estimation paramétrique pouvant être exprimés sous la forme de problèmes moindres carrés généralisés. Les problèmes d'estimation sont: expérience reproductible avec un modèle linéaire généralisé, groupement d'équations, et estimation dans un système dynamique. Nous avons présenté des algorithmes numériquement stables pour la résolution de problèmes de ce genre et nous avons effectué une comparaison avec les méthodes existantes qui sont numériquement instables. Ces algorithmes ont été développés conjointement avec C. Paige, faisant suite au travail déjà entrepris par celui-ci [18].

ACKNOWLEDGEMENT

I would like to thank Professor C. Paige, not only for his invaluable supervision of the work reported here but also for all that I have learned from him about computational linear algebra. It is always a pleasure to work with him and I am grateful to him for his guidance and encouragement during the entire period of the work.

I also thank Mrs. Anneli Hogsden for typing this thesis.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 Overview.

Estimation is a process of extracting information concerning a parameter or a vector of parameters from experimental data. The concepts of least squares estimation and curve fitting were introduced in the early 1800's by Legendre and Gauss mainly for the purpose of reducing physical and astronomical data. However, many major contributions to the field have been made in recent years. Because of the accessibility of computers and the development of numerical techniques, several old problems have been reformulated in a setting appropriate for obtaining efficient numerical solutions.

Estimation theory gradually found its way into many disciplines of science and engineering. Today, the extent to which it has influenced a variety of subjects can be felt by enumerating some of the many seemingly unrelated areas of applications such as satellite orbit determination, mathematical modelling of human operators, optimal and adaptive control, determination of radar range, and economic models of supply and demand.

There is a large body of literature available on the applications of estimation theory in different engineering, econometric and other problems. (See for example Grove et al [9], Sage and Melsa [21], Nahi [14], Johnston [11], Theil [22]).

A mathematical model representing a physical system contains a number of equations and each equation contains a number of parameters. The accuracy of the resulting estimate is generally degraded by the combination of modelling and measurement errors. It is practically impossible to include all the affecting parameters in the mathematical model designed to represent a physical system accurately, though usually only a few parameters will have a large effect on the model. Also there is every likelihood of measurement errors at the time of taking observations. All these errors could be grouped together and the resulting effect called "noise" of the model, which could then be treated as a random unknown variable.

In many cases, the so called linear model is often used to express a relationship. A linear model is defined as an equation in random variables and parameters which is linear in the random variables and parameters. In this thesis an attempt will be made to solve different types of linear models efficiently using computers.

The initial specification of the relationship must include some assumptions about the probability distribution of the random noise vector. Different assumptions will give rise to different computational or statistical problems. We will consider two main types of linear models viz. ordinary linear model and general linear model, arising out of the assumptions one makes regarding the noise vector. Details will be given in section 1.3.

In this Chapter we will establish first our notational conventions. A review of the different types of linear models that we will treat will also be given. In assessing the effectiveness of the various algorithms, we will be concerned with the following attributes tentatively listed in decreasing order of importance: generality, stability, accuracy, efficiency and storage requirement. A brief description of these attributes will also be given here. Most of the numerical methods described in this thesis are based, in some way, upon the properties of orthogonal matrices. Givens plane rotations and Householder transformations are often used. We will describe these transformations briefly in this Chapter.

In Chapter 2 we will derive and discuss the established least squares methods of solving the various types of linear model problems. We will also consider three problems of parameter estimation and the existing methods usually used to solve them. Comments on their effectiveness will also be made.

Chapter 3 contains a fast stable algorithm to solve the least squares problem for general linear models. This algorithm is based on the work carried out by Paige [18].

In estimating parameters in many general linear models (Paige [16]; Theil [22], pp. 294-299), it is necessary to find the minimum 2-norm solution of an underdetermined system (e.g. when the number of parameters to be estimated is more than the number of equations). An algorithm will be presented in Chapter 4 to solve some such systems. The algorithm is based on the particular structure of certain systems.

.../4

Chapters 5,6 and 7 contain new numerically stable methods of solving the three different problems of parameter estimation introduced in Chapter 2. A comparison of efficiencies with the existing methods of solution will also be given.

In Chapter 8 we will comment on the methods which are developed in this thesis. We will also comment on the scope of further developments of these methods.

1.2 Notation.

$E(\cdot)$ will denote the expected value and superscript T will denote the transpose of a matrix. Otherwise, capital italic letters will denote matrices, with the symmetric capitals A, H, M, U, V, W, X, Y reserved for symmetric nonnegative definite matrices. A symmetric matrix is said to be nonnegative definite if all its eigenvalues are greater than or equal to zero and one or more could be zero. We also reserve R for denoting upper triangular matrix and L for lower triangular matrix. Letters P and Q will be reserved for denoting orthogonal matrices. Lower case italics will denote column vectors, except for indices i, j, k, l, m, n, s, t. Lower case Greek letters are used for scalars only.

Also we will use $\|\cdot\|$ to denote the 2-norm of a matrix or a vector.

1.3 Linear models.

A univariate linear model can be represented in matrix notation as

$$y = Cx + u \qquad (1.1)$$

where y is a given m-vector, C is a given $m\times n$ matrix, x is the unknown nonstochastic n-vector of parameters and u is a column m-vector of unobservable random noise variables. This specification implies that the dependent variable y is understood to be a random variable which is on the one hand linearly

.../6

related to x and on the other hand determined by chance.

Generally, in a linear model, the number of observations exceeds the number of parameters to be estimated. Therefore, m is greater than n in most of the cases. We will always consider $m \geq n$ unless otherwise stated.

In order to have a meaningful linear model, one must have some initial assumptions about the random noise vector. It is generally assumed that the mathematical expectation of the elements of the random vector is zero. That is,

$$E(u) = 0 . \qquad (1.2)$$

The elements of the noise vector may be uncorrelated. If we also assume that all the elements of the noise vector have the same variance σ^2, which is unknown, the variance-covariance matrix of the vector u can be written as

$$E(uu^T) = \sigma^2 I \text{ , } I \text{ is an identity matrix of order } m. \qquad (1.3)$$

A linear model given in (1.1) with assumptions (1.2) and (1.3) is known as the ordinary linear model (Johnston [11]) .

The assumption that the disturbances are uncorrelated is not always realized, especially when we deal with time series. If the elements of the random vector are correlated, then the variance-covariance matrix of the vector u becomes.

$$E(uu^T) = \sigma^2 W \qquad (1.4)$$

where σ^2 is an unknown parameter and W is a symmetric nonnegative definite matrix of order m . In certain cases W could be singular. When W is singular, there are some restrictions on the dependent variable y , which may be examined to make sure that there is no obvious inconsistency in the model (Rao [20], pp.297).

A linear model (1.1) together with assumptions (1.2) and (1.4) is known as the general linear model (Johnston [11]).

The assumption (1.4) is considerably weaker than assumption (1.3) because it allows unequal diagonal elements of W (heteroscedasticity) as well as for positive and negative correlations of the disturbances (non-zero off diagonal elements).

In principle, an investigator is completely free in his choice of an estimator for x . Several estimation procedures are possible, for instance, the analogy method, the least squares method, the maximum likelihood method, the minimal chi square method etc. Naturally, the choice of the solution procedure depends on the properties of the respective estimators. The properties and the applicability of different methods are determined by the assumptions one is willing to make about the vector of random variables u and it is essentially these assumptions which determine the estimation procedure to be used. In this thesis we are mainly interested in computing efficiently the least squares estimates and to some extent the maximum likelihood estimates of different types of general linear models.

.../8

Statistical properties of the least squares estimates of ordinary and general linear models have been described in detail by Rao [20] , Theil [22], Johnston [11], Golub & Styan [8] etc. If the elements of the noise vector are assumed to be normally distributed, the maximum likelihood estimators and the least squares estimators are the same. Partly because of this we have given more stress to computing least squares estimates of a general linear model.

If we consider the matrix C , given in (1.1) , as a matrix of observed values, the columns of the matrix C may not be linearly independent because of dependent parameters considered in the model. Also the covariance matrix W can be singular when the disturbances are linearly dependent (Theil [22], pp.274-275). Most of the methods that are described in the literature fail when the columns of the matrix C are linearly dependent and W is singular. Golub [7], Businger and Golub [3], Golub and Styan [8] obtained the least squares estimate of ordinary linear models for a general C using orthogonal transformations. Rao [20] obtained the least squares estimate of a general linear model for general C and W using generalized inverses. Paige [17] reformulated the general linear model differently and obtained the least squares estimate of the problem for general C and W using orthogonal transformations.

1.4 Attributes of an efficient algorithm.

We will determine the effectiveness of an algorithm by considering the following attributes viz. generality, stability, accuracy, efficiency and storage requirement.

Generality means that the method is applicable to wide classes of matrices. For example, in our case, a method which works only for nonsingular matrices will not be highly regarded.

A computer performs basic operations viz. addition, subtraction, multiplication and division with rounding errors. The error is dependent on the precision of the computer one is working with. Often, when a problem is solved using a computer, the result we get can be regarded as the solution of a perturbed problem. An algorithm is stable if it yields a solution that is near the exact solution of a slightly perturbed problem. This does not mean that the answers will be accurate. The accuracy of the solution depends on the conditioning of the problem. A problem can be well conditioned or ill conditioned. If a small change in the data results in large change in the solution then the problem is ill conditioned, otherwise it is well conditioned.

Efficiency is measured by the amount of computer time required to solve a particular problem. In estimating the time required by matrix computations, it is traditional to estimate the time required by the multiplication or division and then increase it by some factor to account for other operations. Generally, we will consider 1 operation as one which involves 1 multiplication or 1 division with 1 addition or 1 subtraction.

.../10

Therefore, we can say that the multiplication of two $m \times m$ matrices involves m^3 operations.

1.5 Givens and Householder matrices.

The most common application of orthogonal matrices in numerical analysis is equivalent to the reduction of a given n-vector z to a multiple of the first column vector of the identity matrix, i.e. find an orthogonal $n \times n$ matrix Q such that

$$Q^T z = \pm \gamma e_1, \ \gamma = \| z \| \qquad (1.5)$$

where e_1 is an n-vector of each component zero except for the first one which is 1 . The reason for preferring orthogonal transformations over others is that they do not change the condition of the problem. The reduction given in (1.5) can be done by either a sequence of plane rotation (Givens) matrices or a single elementary orthogonal (Householder) matrix.

Givens matrix is defined as

$$\begin{bmatrix} \alpha & \beta \\ \beta & -\alpha \end{bmatrix} . \qquad (1.6)$$

The choice of α and β to perform the reduction

$$\begin{bmatrix} \alpha & \beta \\ \beta & -\alpha \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \begin{bmatrix} +\gamma \\ 0 \end{bmatrix}$$

is given by

$$\left.\begin{aligned}\gamma^2 &= \xi_1^2 + \xi_2^2 \ , \\ \alpha &= \xi_1/\gamma \text{ and } \beta = \xi_2/\gamma .\end{aligned}\right\} \qquad (1.7)$$

When an n-vector z is reduced by Givens rotation matrices to a multiple of the first column of the identity matrix, then (n-1) rotations are needed. Every rotation will zero out one element and adjust another element such that the updated $\|z\|$ is preserved. Thus, Givens plane rotation preserves the size of the vector.

To perform the same reduction in one step using a single Householder matrix, we can form

$$Q = I - \frac{1}{\alpha} uu^T \qquad (1.8)$$

where

$$\left.\begin{aligned}u &= z + \gamma e_1 \\ \alpha &= \frac{1}{2} \|u\|^2 \\ \gamma &= \text{sign}(z^T e_1)\|z\|\end{aligned}\right\} \qquad (1.9)$$

such that

$$Q^T z = -\gamma e_1 \ .$$

From (1.9) it can be seen that the Householder transformation also preserves the size of z .

.../12

The choice of using orthogonal Givens rotations or Householder transformations depends on the type of problem one is working with. There are cases when one is better than the other. In the general case, the Householder transformations only involve about half the number of multiplications required for Givens rotations. Gentleman [5] and Hammarling [10] have shown that it is possible to implement square root free versions of Givens rotation in about half the number of multiplications of the classical method.

CHAPTER 2

CLASSICAL DERIVATION AND SOLUTION OF VARIOUS TYPES OF LINEAR MODEL PROBLEMS.

2.1 Introduction.

In this Chapter we will derive and discuss the established least squares methods of solving two main types of linear models viz. ordinary linear model and general linear model. We will then describe the three parameter estimation problems which we will consider in this thesis. Existing methods of solution of these problems will also be discussed and comments will be made on their effectiveness.

2.2 Ordinary least squares problem.

We know that equation (1.1) in Chapter 1 together with the assumptions (1.2) and (1.3) constitute an ordinary linear model, i.e. an ordinary linear model can be described as

$$y = Cx + u \; ; \quad E(u) = 0 \; , \quad E(uu^T) = \sigma^2 I \; . \qquad (2.1)$$

The problem of obtaining least squares estimator for an ordinary linear model is known as the ordinary least squares problem (Johnston [11]).

2.2.1 Derivation of the problem.

Let $\hat{x}$ be an estimate of (2.1). Then $(y - C\hat{x})$ is the vector of m residuals. The ordinary least squares problem is to find x that minimizes

$$(y - Cx)^T(y - Cx) \tag{2.2}$$

or

$$\hat{x} = \arg\min_x \|y - Cx\| \tag{2.3}$$

where this notation is short for "$\hat{x}$ is the argument that minimizes $\|y - Cx\|$ with respect to x". For a given value of y, the vector $\hat{x}$ that solves the problem (2.3) is called the ordinary least squares estimate of x.

2.2.2 Method of solution.

In many places in the numerical, engineering, econometric and statistical literatures, the ordinary least squares problem has been solved by forming normal equations. Differentiating (2.2) with respect to x and equating to zero we get

$$C^TC\,x = C^T y \,. \tag{2.4}$$

Hence if C has linearly independent columns, the least squares estimator is given by

$$\hat{x} = (C^TC)^{-1} C^Ty \tag{2.5}$$

which minimizes (2.2) since C^TC is positive definite.

We know that C can have linearly dependent columns. In that case C^TC becomes singular and the method fails. Moreover the condition number for the solution of equations in (2.4) is the square of the condition number of C, and (2.4) can have much worse condition than the original least squares problem. Thus the method of estimating $\hat{x}$ by forming normal equations is not numerically stable (de Jong [4]).

The ordinary least squares problem for a general matrix C has been solved successfully by Businger and Golub [3] and Golub [7] using orthogonal transformations, following Householder.

First we can choose an orthogonal matrix Q such that

$$Q^T C = \begin{bmatrix} Q_1^T C \\ Q_2^T C \end{bmatrix} = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad Q^T = \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} \qquad (2.6)$$

where R is a full row rank matrix and Q is partitioned so that the number of rows in Q_1^T is the same as that of R .

Since the 2-norm is unaffected by orthogonal transformations, (2.3) can be written as

$$\hat{x} = \arg\min_x \left\| \begin{bmatrix} Q_1^T y \\ Q_2^T y \end{bmatrix} - \begin{bmatrix} R \\ 0 \end{bmatrix} x \right\| \qquad (2.7)$$

from which it follows that $\hat{x}$ satisfies

$$R\hat{x} = Q_1^T y \qquad (2.8)$$

and the residue of the solution is given by

$$\| Q_2^T y \| \; . \qquad (2.9)$$

Since R has full row rank, (2.8) is solvable for $\hat{x}$. If R is square $\hat{x}$ is unique, otherwise we will have many $\hat{x}$ satisfying (2.8). We are, generally, concerned with the minimum 2-norm of such $\hat{x}$. In this case R will be in upper trapezoidal form. So we can find an orthogonal matrix P such that

$$RP = (0,\bar{R}) \quad , \quad P = (P_1,P_2) \tag{2.10}$$

where $\bar{R}$ is a non singular upper triangular matrix. (2.8) can thus be transformed to

$$(0,\bar{R})P^T\hat{x} = Q_1^T y \tag{2.11}$$

Let

$$z = \begin{bmatrix} z_1 \\ \\ z_2 \end{bmatrix} = P^T\hat{x} \quad . \tag{2.12}$$

We can now solve

$$\bar{R}z_2 = Q_1^T y \tag{2.13}$$

for z_2 and setting $z_1 = 0$, the minimum norm least squares estimator for the model in (2.1) is given by

$$\hat{x} = P_2 z_2 \quad . \tag{2.14}$$

.../17

2.2.3 Operation count.

In counting the number of operations required to solve an ordinary least squares problem we will assume for simplicity that the matrix C of dimension m×n has full column rank.

To compute (2.6) using Householder transformations takes about

$$mn^2 - \frac{n^3}{3} \quad . \qquad (2.15)$$

operations. If we use 4 multiplication Givens plane rotations, the total number of operations required to compute (2.6) is

$$2mn^2 - \frac{2}{3}n^3 \quad . \qquad (2.16)$$

If we use fast square root free Givens rotation (Gentleman [5], Hammarling [10]), the total number of operations required to compute (2.6) is given by

$$mn^2 - \frac{n^3}{3} \quad . \qquad (2.17)$$

Thus we see that there is no advantage in choosing one transformation over the other. But if the matrix C is a large sparse matrix, then the use of Givens plane rotations has a definite advantage over Householder transformations.

2.3 Generalized least squares problem.

The system of linear equations (1.1) with assumptions (1.2) and (1.4) given in Chapter 1 represents a general linear model, i.e. a general linear model can be written as

$$y = Cx + u \quad ; \quad E(u) = 0, \; E(uu^T) = \sigma^2 W \; . \qquad (2.18)$$

The problem of obtaining least squares estimator for a general linear model is known as the generalized least squares problem (Johnston [11]).

2.3.1 Derivation of the problem.

Like the ordinary least squares problem, methods of solving generalized least squares problems are widely available in the literature ([11], [13]). We will first assume that W is a symmetric positive definite matrix. Then we can carry out the Cholesky decomposition of W such that

$$W = LL^T$$

where L is a lower triangular matrix.

Multiplying both sides of (2.18) by L^{-1} we get

$$L^{-1}y = L^{-1}Cx + v \; ; \; v = L^{-1}u \, , \; E(v) = 0 \; , \; E(vv^T) = \sigma^2 I. \quad (2.19)$$

Thus (2.19) becomes an ordinary linear model. Therefore, from (2.19) we see that the generalized least squares problem becomes: find x that minimizes

$$(y - Cx)^T W^{-1} (y - Cx) \quad . \qquad (2.20)$$

This form was originally proposed by Aitken [1]. The vector $\hat{x}$ that minimizes (2.20) is called the least squares estimate of (2.18).

2.3.2 Method of solution.

Often the generalized least squares problem has been solved, like ordinary least squares problem, by forming the normal equations

$$C^TW^{-1}Cx = C^TW^{-1}y \tag{2.21}$$

and consequently, if C has full column rank

$$\hat{x} = (C^TW^{-1}C)^{-1}C^TW^{-1}y \tag{2.22}$$

is the least squares estimate of the general linear model given in (2.18).

Like the ordinary least squares problem, this method of solution fails when C has linearly dependent columns or W is singular.

From (2.19) we see that the generalized least squares problem can also be solved by solving the following ordinary least squares problem:

$$\hat{x} = \arg\min_{x} \|L^{-1}y-L^{-1}Cx\| . \tag{2.23}$$

Now (2.23) can be evaluated by applying the stable method used for solving ordinary least squares problems. But the difficulty with the problem (2.23) is that it does not work when W is singular or near singular. If W is ill conditioned $\|L^{-1}\|$ will be large and therefore the method will introduce unnecessary errors. Often since $\|W^{-1}\| = \|L^{-1}\|^2$ the situation

.../20

is much worse for (2.21) and (2.22). Björck [2] has designed a method to handle less than full rank L. His method does not work well when L has full rank but is poorly conditioned and therefore leads to the same unnecessary numerical inaccuracies suffered by the methods directly based on (2.23). To avoid both the difficulty caused by singularity and that caused by ill condition, the following formulation was proposed by Paige [17]:

$$\underset{v,x}{\text{minimize}} \ v^T v \quad \text{subject to} \quad y=Cx+Bv \qquad (2.24)$$

where $W = BB^T$ is the Cholesky decomposition. If B is square lower triangular then (2.24) is mathematically equivalent to (2.23) with B and L the same. Now the formulation allows all C and B in a compatible system. B is non square when the variance-covariance matrix W of the noise term is singular. The Cholesky decomposition is still possible for the symmetric nonnegative definite matrix W (Lawson and Hanson [13] pp. 124). It can be shown that the approach (2.24) gives the same answers as Rao's unified theory of linear estimation [20] and is in a form that leads directly to good computational algorithms.

It is now possible to solve a generalized least squares problem given in the form (2.24) using orthogonal transformations (Paige [17]).

We can find an orthogonal matrix Q such that

$$Q^T C = \begin{bmatrix} R \\ 0 \end{bmatrix} \quad , \quad Q^T = \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} \quad . \tag{2.25}$$

where R is a full row rank matrix and Q^T is partitioned so that Q_1^T has the same number of rows as that of R . Thus the constraints in (2.24) become

$$\begin{bmatrix} Q_1^T y \\ Q_2^T y \end{bmatrix} = \begin{bmatrix} R \\ 0 \end{bmatrix} x + \begin{bmatrix} Q_1^T B \\ Q_2^T B \end{bmatrix} v \quad . \tag{2.26}$$

Once v is known, one can easily solve

$$Q_1^T y = Rx + Q_1^T Bv \tag{2.27}$$

for x .

Therefore, the problem given in (2.24) reduces to

$$\underset{v}{\text{minimize}} \quad v^T v \quad \text{subject to} \quad Q_2^T y = Q_2^T Bv \tag{2.28}$$

which is nothing but finding the minimum 2-norm solution of the underdetermined system

$$Q_2^T Bv = Q_2^T y \quad . \tag{2.29}$$

We also solve (2.29) by using orthogonal transformations. We can form an orthogonal matrix P such that

$$Q_2^T BP = [\,0,S\,]\;,\;\; P = [P_1, P_2] \qquad (2.30)$$

where S has a full column rank and P is partitioned so that the number of columns of P_2 is the same as that of S.

Let

$$w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = P^T v = \begin{bmatrix} P_1^T v \\ P_2^T v \end{bmatrix} . \qquad (2.31)$$

Then solving

$$Sw_2 = Q_2^T y \qquad (2.32)$$

for w_2 and letting $w_1 = 0$, the minimum norm solution of (2.29) is given by

$$\hat{v} = P_2 w_2 \quad . \qquad (2.33)$$

Since S has full column rank, w_2 is unique if the set of equations is consistent and so $\hat{v}$ is unique. If S is not square, we can find an orthogonal matrix $\tilde{Q}$ such that

$$\tilde{Q}^T S = \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix} \;,\;\; \tilde{Q}^T = \begin{bmatrix} \tilde{Q}_1^T \\ \tilde{Q}_2^T \end{bmatrix} \qquad (2.34)$$

where $\tilde{R}$ is a nonsingular matrix.

We can now solve

$$\tilde{R} w_2 = \tilde{Q}_1^T Q_2^T y \qquad (2.35)$$

.../23

for w_2 . We can check the consistency of the model by checking $\|\tilde{Q}_2^T Q_2^T y\|$ which should be very small for a consistent model. Thus a check can be provided on the correctness of the model.

Therefore, knowing $\hat{v}$ given in (2.33), we can obtain $\hat{x}$ from equation (2.27) which will be the least squares estimate of the general linear model given in (2.18).

Kourouklis [12] programmed the algorithm for general C and W developed by Paige [17] in ALGOLW for the IBM 370 system. The results, checked against the IMSL subroutine LLSQAR, are accurate to machine accuracy. He also compared these results with the results obtained by using (2.23). He found that, for ill conditioned L , they differ from the first or second significant digit.

2.3.3 Operation count.

For simplicity, we will assume that the matrix C is a full column rank matrix of dimension m×n and B is an m×m non singular lower triangular matrix. We will obtain operation counts for the case when the problem is solved by Householder transformations and also by using Givens plane rotations.

To reduce C to R using Householder transformations takes about

$$mn^2 - \frac{1}{3} n^3 \qquad (2.36)$$

operations.

.../24

Forming Q^TB takes about

$$2m^2n - mn^2 \tag{2.37}$$

operations and then reducing Q^TB to lower triangular form takes

$$\frac{2m^3}{3} \tag{2.38}$$

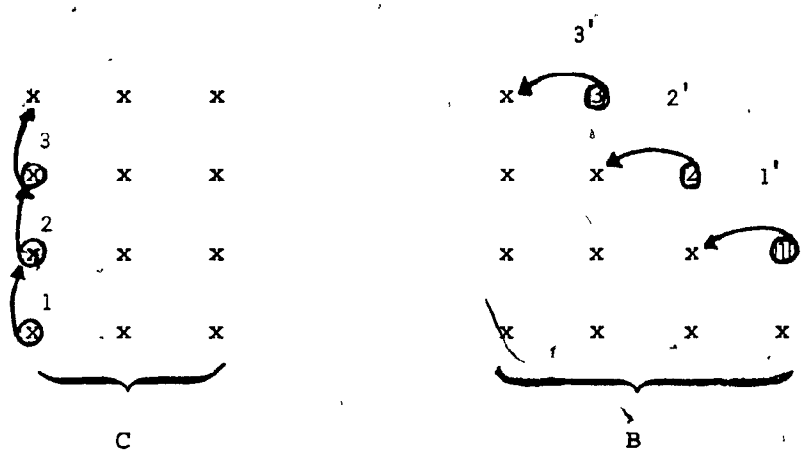
operations.

Since the other operations are relatively smaller, the total operations required to solve a generalized least squares problem using Householder transformations is about

$$\frac{2m^3}{3} + 2m^2n - \frac{1}{3}n^3 \ . \tag{2.39}$$

When we used Householder transformations at the time of reducing C to an upper triangular matrix, we let the lower triangular form of B be destroyed. This is the disadvantage of using Householder transformations in this way for solving a generalized least squares problem.

We can also reduce C to an upper triangular form using Givens rotations. We apply the rotations in such a way that we eliminate the elements below the leading diagonal elements of C while maintaining the lower triangular form of B . For example with m = 4, n = 3, the initial step will be



(2.40)

The rotations are ordered 1, 1', 2, 2', 3, 3' and the non-zero element $\boxed{i}$, introduced by rotation i from the left, is immediately made zero by rotation i' from the right. We continue like this until C is reduced to the upper triangular form. Kourouklis [12] programmed this approach.

If we use 4 multiplication rotation, total number of operations required to reduce C to the upper triangular form and at the same time maintaining the form of B throughout is

$$4m^2n - \frac{2}{3}n^3 . \qquad (2.41)$$

From (2.39) and (2.41) we see that for an overdetermined system the method using Givens rotations is more efficient than that using Householder transformations because of the term m^3 in (2.39). If the square root free rotation is used, Givens rotations are always economical to use. Therefore, it is possible to say that in the general case Givens rotations should be used to solve a generalized least squares problem which is presented in the form given in (2.24) in preference to the Householder transformations.

2.4 Parameter estimation problem.

In this section we will introduce three problems of parameter estimation which we intend to solve efficiently in this thesis.

2.4.1 A repeatable experiment with a general linear model.

We know that a general linear model can be written as

$$y = Cx + Bv \; ; \; E(v) = 0, \; E(vv^T) = \sigma^2 I \qquad (2.42)$$

where y is a known m-vector, C is a known m×n matrix, v is the unknown k-dimensional noise vector and B is a known m×k matrix with full column rank.

In some situations such as laboratory experiments, in order to get a reliable estimate, one can make repeated experiments with the same model under the same set of conditions. Then y and v will be different for different experiments while C and B remain the same. If we assume that the experiments are independent of each other, the model given in (2.42) will be of the form

$$y_i = Cx+Bv_i, \; i=1,2,\ldots,t; \; E(v_i) = 0, \; E(v_i v_j^T) = \delta_{ij}\sigma^2 I \qquad (2.43)$$

where δ_{ij} is the Kronecker delta.

The t different linear models in (2.43) can be grouped together and written as

$$\bar{y} = \bar{C}x + \bar{B}\bar{v} \; ; \; E(\bar{v}) = 0 \; , \; E(\bar{v}\bar{v}^T) = \sigma^2 I \qquad (2.44)$$

where

$$\underset{tm\times 1}{\bar{y}} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ y_t \end{bmatrix}, \quad \underset{tm\times n}{\bar{C}} = \begin{bmatrix} C \\ C \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ C \end{bmatrix}, \quad \underset{tm\times tk}{\bar{B}} = \begin{bmatrix} B & & & & \\ & B & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & B \end{bmatrix}, \quad \underset{tk\times 1}{\bar{v}} = \begin{bmatrix} v_1 \\ v_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ v_t \end{bmatrix} \qquad (2.45)$$

For large m and t, the dimensions of the problem given in (2.44) become very large. Hence the previous method of solving a generalized least squares problem, when applied to (2.44), will take a lot of storage and unneccesary computations giving an inefficient algorithm. It is also seen that only $\bar{y}$, C, B need to be known inorder to know the entire system.

If in a controlled experiment each set of observations is taken with a different set of measuring instruments, then it is possible to have different variance-covariance matrices for the noise term. Let W_i be the variance-covariance matrix of the noise vector for the i th observation vector y_i. Also let $W_i = B_i B_i^T$ be the Cholesky decomposition, where B_i is $m\times k_i$ matrix with full column rank. Then $\bar{B}$ in (2.45) becomes

$$\underset{tm\times \sum_{i=1}^{t} k_i}{\bar{B}} = \begin{bmatrix} B_1 & & & & \\ & B_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & B_t \end{bmatrix} . \qquad (2.46)$$

In Chapter 5, we present an algorithm for solving such problems with block diagonal $\bar{B}$.

2.4.2 Grouping of Equations.

Another application of the generalized least squares problem occurs in the estimation of parameters of a group of ordinary linear models whose noise vectors are correlated (Zellner [23]). The idea is to estimate the parameters of the different sets of equations jointly by utilizing the relationship among the disturbances of each set of equations. Zellner [23] has shown that when different sets of "independent" variables appear in equations of the system and when there exist correlations between the noise terms, the generalized least squares estimators are asymptotically more efficient than those obtained by the application of ordinary least squares to each set of equations in turn.

Suppose that the i th equation in a group of n sets is

$$y_i = C_i x_i + u_i \ , \quad i=1,2,...,n \qquad (2.47)$$

where y_i is an m-vector, C_i an $m \times k_i$ matrix, u_i is an m dimensional random noise vector. All the sets of equations can be grouped together and can be written as

$$y = Cx + u \qquad (2.48)$$

where

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_n \end{bmatrix}, \; C = \begin{bmatrix} C_1 & & & \\ & C_2 & & \\ & & \ddots & \\ & & & C_n \end{bmatrix}, \; x = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_n \end{bmatrix}, \; u = \begin{bmatrix} u_1 \\ u_2 \\ \cdot \\ \cdot \\ u_n \end{bmatrix} . \quad (2.49)$$

The random noise vector u has a zero mean and its variance-covariance matrix is given by

$$W = E(uu^T) = \begin{bmatrix} E(u_1u_1^T) & E(u_1u_2^T) & \cdots & E(u_1u_n^T) \\ E(u_2u_1^T) & E(u_2u_2^T) & \cdots & E(u_2u_n^T) \\ \cdot & \cdot & \cdot\;\cdot & \cdot \\ \cdot & \cdot & \cdot\;\cdot & \cdot \\ \cdot & \cdot & \cdot\;\cdot & \cdot \\ E(u_nu_1^T) & E(u_nu_2^T) & \cdot\;\cdot & E(u_nu_n^T) \end{bmatrix} \quad (2.50)$$

where $E(u_iu_j^T)$ is the variance-covariance matrix for the noise vectors of the i th and the j th sets of equations. By assumption

$$E(u_iu_j^T) = \sigma_{ij}I \; , \quad i,j = 1,2,..,n \qquad (2.51)$$

where I is a unit matrix of order m . Therefore,

$$W = \begin{bmatrix} \sigma_{11}I & \sigma_{12}I & \cdot\;\cdot & \sigma_{1n}I \\ \sigma_{21}I & \sigma_{22}I & \cdot\;\cdot & \sigma_{2n}I \\ \cdot & \cdot & \cdot\;\cdot & \cdot \\ \cdot & \cdot & \cdot\;\cdot & \cdot \\ \sigma_{n1}I & \sigma_{n2}I & \cdot\;\cdot & \sigma_{nn}I \end{bmatrix}$$

$$= \quad S \otimes I \; , \quad S = (\sigma_{ij})$$

.../30

where $\otimes$ is the Kronecker product.

The general linear model, thus, becomes

$$y = Cx+u \; ; \; E(u) = 0 \; , \; E(uu^T) = W \; . \tag{2.52}$$

2.4.2.1 Existing method of solution.

Zellner [23] solved this problem by using Aitken's generalized least squares method, i.e. by forming the normal equations and therefore the least squares estimate of the model is given by

$$\hat{x} = (C^T W^{-1} C)^{-1} C^T W^{-1} y \; . \tag{2.53}$$

Let

$$S^{-1} = (\sigma^{ij})$$

so that

$$W^{-1} = S^{-1} \otimes I \; . \tag{2.54}$$

Then (2.53) can be written as

$$\hat{x} = A^{-1} b \tag{2.55}$$

.../31

where

$$A = \begin{bmatrix} \sigma^{11}C_1^TC_1 & \sigma^{12}C_1^TC_2 & \cdots & \sigma^{1n}C_1^TC_n \\ \sigma^{21}C_2^TC_1 & \sigma^{22}C_2^TC_2 & \cdots & \sigma^{2n}C_2^TC_n \\ \vdots & \vdots & & \vdots \\ \sigma^{n1}C_n^TC_1 & \sigma^{n2}C_n^TC_2 & \cdots & \sigma^{nn}C_n^TC_n \end{bmatrix}, b = \begin{bmatrix} \sum_{j=1}^{n} \sigma^{1j}C_1^Ty_j \\ \sum_{j=1}^{n} \sigma^{2j}C_2^Ty_j \\ \vdots \\ \sum_{j=1}^{n} \sigma^{nj}C_n^Ty_j \end{bmatrix}. \quad (2.56)$$

(2.55) can best be evaluated by applying Cholesky decomposition. The matrix A is symmetric positive definite matrix of rank $\sum_{i=1}^{n} k_i$. Therefore, A has a Cholesky factorization of the form

$$A = LL^T \quad (2.57)$$

where L is a lower triangular matrix. Hence, first solving

$$Ly = b \quad (2.58)$$

for y, we can estimate $\hat{x}$ by solving

$$L^T\hat{x} = y \quad (2.59)$$

for $\hat{x}$.

Zellner's method makes effective use of S^{-1} to obtain the generalized linear least squares estimators. As we have discussed earlier, W can be singular or C_i may have linearly dependent columns. Then Zellner's method does not work. Also we know that by squaring a matrix its condition number is also squared which results in unnecessary errors in the solution. We also know that squaring matrices on a computer can result in loosing information unnecessarily (see for example [7]).

2.4.2.2 Operation count.

While counting the number of operations for solving a generalized least squares problem using Zellner's method, we will assume that matrices C_i for $i = 1,2,...,n$ are of equal dimension $m \times k$.

To form A takes about

$$\frac{1}{2} mn^2k^2 \qquad (2.60)$$

operations. Number of operations necessary for the Cholesky decomposition of A is

$$\frac{1}{6} n^3k^3 . \qquad (2.61)$$

Since the other operations are relatively smaller we can say that Zellner's method of solving the problem (2.52) takes about

$$\frac{1}{2} mn^2k^2 + \frac{1}{6} n^3k^3 \qquad (2.62)$$

operations.

2.4.3 Estimation in a dynamical system.

The problem of parameter estimation in a nonlinear dynamical system occurs in many engineering fields where it often involves the processing of large amounts of data quickly and accurately. Grove et al [9] have used a maximum likelihood parameter estimation procedure for estimating the stability and control parameters from the flight test data of aircrafts.

.../33

Let us consider a simple form of parameter estimation problem in a dynamical system. Let the mathematical model of a dynamic system be of the form

$$\dot{x} = f(x,p,t) \qquad (2.63)$$

where x is the state vector of m elements, p is the parameter vector to be estimated and t represents time.

Let us observe this dynamical system at discrete times

$$0 = t_1 < t_2 < . . . < t_k . \qquad (2.64)$$

Let x_i be the observed state vector of the system at time t_i. Now x_i is also some function of the vector p which is unknown. But the measurement of the state vector will be polluted with noise. We assume the measured state vector $\tilde{x}_i$ can be written as

$$\tilde{x}_i = x_i+v_i \ ; \ E(v_i) = 0, \ E(v_i v_j^T) = \delta_{ij}V, \ i,j=1,2,...,k \qquad (2.65)$$

where x_i is the true state vector, v_i is the noise vector during the observation at time t_i, V is the variance-covariance matrix (assumed to be positive definite matrix) of the unknown noise vector and δ_{ij} is the Kronecker delta.

The maximum likelihood estimates of p and V are obtained by maximizing the likelihood function $\phi(p,V)$ with respect to p and V.

We will assume that the elements of the noise vector are distributed normally. Then the likelihood function $\phi(p,V)$ is given by

.../34

$$\phi(p,V) = \frac{1}{(2\pi)^{mk/2}\det(V)^{k/2}} \cdot \exp\left\{-\frac{1}{2}\sum_{i=1}^{k}(\tilde{x}_i - x_i)^T V^{-1}(\tilde{x}_i - x_i)\right\} . \qquad (2.66)$$

Since this is always positive and the logarithm is a monotone increasing function for positive real arguments, to maximize $\phi(p,V)$, we may as well maximize

$$\begin{aligned} \phi^*(p,V) &= \frac{2}{k}\ \ell n\ \phi(p,V) + m\ \ell n\ 2\pi \\ &= -\ell n\ \det(V) - \frac{1}{k}\sum_{i=1}^{k}(\tilde{x}_i - x_i)^T V^{-1}(\tilde{x}_i - x_i) \\ &= -\ell n\ \det(V) - \frac{1}{k}\ \text{trace}\ (Z^T V^{-1} Z) \qquad (2.67) \\ &= -\ell n\ \det(V) - \frac{1}{k}\ \text{trace}\ (V^{-1} Z\, Z^T)\ ,\ \text{since} \end{aligned}$$

trace (AB) = trace (BA) for compatible dimensions

where

$$Z = [z_1, z_2, \ldots, z_k]\ ,\ z_i = \tilde{x}_i - x_i\ . \qquad (2.68)$$

It can be shown that the value of V which maximizes $\phi^*(p,V)$ i.e. which minimizes $-\phi^*(p,V)$ for a fixed p is given by (see Grove et al [9])

$$V(p) = \frac{1}{k}\ Z\ Z^T = FF^T\ ,\ \text{say}\ \ F = \frac{1}{\sqrt{k}}\ Z\ . \qquad (2.69)$$

It should be noted that k , the number of observations, should be more than the number of elements in the vector x_i in order to have V non singular.

.../35

Similarly, for a fixed V, the value of p that minimizes $-\phi^{*}(p,V)$ is given by

$$p = \text{argument that minimizes } \operatorname{trace}(Z^{T}V^{-1}Z) \quad . \qquad (2.70)$$

From (2.65) and (2.68) we find that this minimization problem becomes: find p that minimizes

$$\sum_{i=1}^{k} \| F^{-1}\tilde{x}_i - F^{-1}x_i(p) \|^{2} \quad . \qquad (2.71)$$

The problem given in (2.71) is different than the generalized least squares problem because F^{-1} is also a function of the parameter p .

2.4.3.1 Existing method of solution.

Some methods of solving this type of estimation problem have been discussed by Paige [16] , Grove et al [9]. Their methods are basically iterative.

For a given p, V in (2.69) is evaluated. Then the linear extrapolation of the vector $x_i(p)$ is taken which is given by

$$x_i(p+\delta p) \simeq x_i(p) + \frac{\partial x_i(p)}{\partial p^{T}} \delta p \quad . \qquad (2.72)$$

This gives the approximation of (2.71): find δp that minimizes

$$\sum_{i=1}^{k} \| F^{-1}(\tilde{x}_i - x_i(p)) - F^{-1} \frac{\partial x_i(p)}{\partial p^{T}} \delta p \|^{2} \qquad (2.73)$$

or

$$\hat{\delta p} = \underset{\delta p}{\arg\min} \| y - C\delta p \|^2 \qquad (2.74)$$

where

$$y = \begin{bmatrix} F^{-1}(\tilde{x}_1 - x_1(p)) \\ F^{-1}(\tilde{x}_2 - x_2(p)) \\ \vdots \\ F^{-1}(\tilde{x}_k - x_k(p)) \end{bmatrix} , \quad C = \begin{bmatrix} F^{-1} \dfrac{\partial x_1(p)}{\partial p^T} \\ F^{-1} \dfrac{\partial x_2(p)}{\partial p^T} \\ \vdots \\ F^{-1} \dfrac{\partial x_k(p)}{\partial p^T} \end{bmatrix} .$$

Thus we find $p+\hat{\delta p}$ which optimizes (2.70) with $V = V(p)$ in (2.69) and the approximation to $x_i(p)$ in (2.72). Next we find the improved value of $V(p+\hat{\delta p})$ in (2.69) and repeat the process until convergence. The algorithm of the problem may be written as (Paige [16]).

i) guess p

ii) form Z in (2.69)

iii) transform $ZP = [L,0]$, L lower triangular, P orthogonal

iv) form $C_i = \dfrac{\partial x_i(p)}{\partial p^T}$, $y_i = \tilde{x}_i - x_i(p)$; $i=1,2,\dots,k$

v) solve for $i=1,2,\dots,k$ $\begin{cases} L\bar{C}_i = C_i \\ L\bar{y}_i = y_i \end{cases}$

to give

.../37

$$C = \begin{bmatrix} \bar{C}_1 \\ \bar{C}_2 \\ \cdot \\ \cdot \\ \bar{C}_k \end{bmatrix} , \quad y = \begin{bmatrix} \bar{y}_1 \\ \bar{y}_2 \\ \cdot \\ \cdot \\ \bar{y}_k \end{bmatrix}$$

vi) find

$$\hat{\delta p} = \arg\min_{\delta p} \| y - C\delta p \|$$

vii) improve $p = p + \hat{\delta p}$ and go to (ii)

It is often the case that steps (v) and (vi) are the most time consuming steps. If the number of observations is large, then the size of the problem is disconcerting. Also if the number of observation k is less than the number of elements in x_i, then V is singular and so L^{-1} does not exist. If V is ill conditioned then L^{-1} is large and therefore, the answers will be inaccurate. Hence the method is not numerically stable for ill conditioned V.

We can combine steps (v) and (vi) in the following general linear model

$$y = C\delta p + v; \; E(v) = 0, \; E(vv^T) = \bar{V} \qquad (2.75)$$

where

$$C = \begin{bmatrix} \frac{\partial x_1(p)}{\partial p^T} \\ \frac{\partial x_2(p)}{\partial p^T} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial x_k(p)}{\partial p^T} \end{bmatrix} \quad \text{and} \quad \bar{V} = \begin{bmatrix} V & & & & \\ & V & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & V \end{bmatrix} .$$

$\bar{V}$ here is a symmetric nonnegative definite matrix.

In Chapter 7, an efficient numerically stable algorithm has been presented to obtain the least squares estimation of a general linear model of the form

$$y = Cx + u; \; E(u) = 0, \; E(uu^T) = U \qquad (2.76)$$

where

$$U = \begin{bmatrix} U_1 & & & & \\ & U_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & U_k \end{bmatrix} .$$

The form (2.75) or (2.76) will allow us to obtain the least squares estimate of (2.76) for a general C and U. The algorithm is a new and numerically stable one which has been devised by C. Paige and the author.

2.4.3.2 Operation count.

We will give an operation count to accomplish the steps (v) and (vi) of the algorithm of Paige [16]. We will assume, for simplicity, that each C_i in C' is of the same dimension m×n and L is lower triangular.

To compute step (v) takes about

$$\frac{1}{2} m^2nk \tag{2.77}$$

operations.

Transformations of C to an upper triangular form by using Householder transformations takes about

$$mn^2k - \frac{1}{3} n^3 \tag{2.78}$$

operations.

Thus to accomplish steps (v) and (vi), it takes about

$$\frac{1}{2} m^2nk + mn^2k - \frac{1}{3} n^3 \tag{2.79}$$

operations.

CHAPTER 3

FAST STEP BY STEP COMPUTATIONS FOR GENERALIZED LEAST SQUARES PROBLEMS.

3.1 Introduction.

We know [17] that a generalized linear least squares problem can be stated as

$$\underset{v,x}{\text{minimize}}\ v^Tv \text{ subject to } y = Cx+Bv;\ E(v) = 0,\ E(vv^T) = \sigma^2 I \qquad (3.1)$$

where y is a given m-vector, C is a given $m\times n$ matrix, x is an unknown n-vector of parameters, v is an unknown random noise term and B is a known matrix with full column rank.

In this Chapter, we present a fast step by step algorithm for solving the problems of type (3.1). This will be based on the work carried out by Paige [18] . We will assume for simplicity that C has full column rank and B is lower triangular although the algorithm works for column deficient C and non-square B . An operation count will also be given on the basis of these assumptions.

3.2 Method of solution.

We have seen in Chapter 2 that the application of Givens plane rotations has definite advantage over Householder transformations in solving a generalized least squares problem when formulated like (3.1). Therefore, we will use Givens plane rotations only in reducing the system.

By applying orthogonal transformations from left and right, it is possible to transform the initial data [y,C,B] in (3.1) as follows:

$$Q^T[\,y,C,B\,]\begin{array}{c}\left[\begin{array}{ccc}1&&\\&I&\\&&P\end{array}\right]\\ \begin{matrix}\underbrace{\quad}_{1}&\underbrace{\quad}_{n}&\underbrace{\quad}_{m}\end{matrix}\end{array} = \begin{array}{c}\left[\begin{array}{ccccc}0&0&L_1&0&0\\ \eta&0&g^T&\rho&0\\ z&R^T&L_{21}&r&L_2\end{array}\right]\\ \begin{matrix}\underbrace{\quad}_{1}&\underbrace{\quad}_{n}&\underbrace{\qquad}_{m-n-1}&\underbrace{\quad}_{1}&\underbrace{\quad}_{n}\end{matrix}\end{array}\begin{array}{l}\}m-n-1\\ \}1\\ \}n\\ \\ \end{array} \qquad (3.2)$$

where Q and P are orthogonal matrices, L_1, L_2 , R^T are non singular lower triangular matrices. η and ρ are scalars. For general C and B , rotations can be so chosen that L_1, L_2 , R will have full column rank.

Writing

$$P^Tv = \begin{bmatrix}v_1\\ \mu\\ v_2\end{bmatrix} \qquad (3.3)$$

it is seen from the constraints in (3.1) and transformations in (3.2) that

$$L_1v_1 = 0 \;, \qquad (3.4)$$

$$\eta = g^Tv_1 + \rho\,\mu \;,\text{ and} \qquad (3.5)$$

$$z = R^Tx+L_{21}v_1+\mu r+L_2v_2 \;. \qquad (3.6)$$

(3.4) implies $v_1 = 0$ since L_1 is lower triangular. Therefore (3.6) can be written as

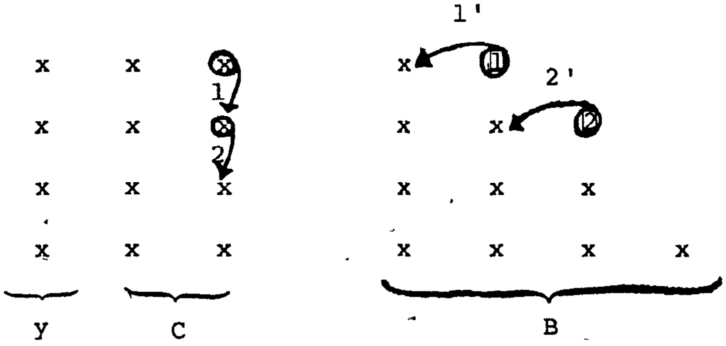
$$z = R^T x + \mu r + L_2 v_2 \ . \tag{3.7}$$

Since R^T has a full row rank, it can always be solved for x irrespective of v_2. So the minimum norm least squares solution $\hat{x}$ of (3.1) is given by

$$\begin{bmatrix} \rho & 0 \\ r & R^T \end{bmatrix} \begin{bmatrix} \mu \\ \hat{x} \end{bmatrix} = \begin{bmatrix} \eta \\ z \end{bmatrix} \ . \tag{3.8}$$

For the constraints to be consistent, ρ has to be nonzero if η is nonzero. If $\rho = 0$ but $\eta \neq 0$, then the constraints in (3.1) are incompatible. Thus, we can have a check on the feasibility of the model with nonsquare B.

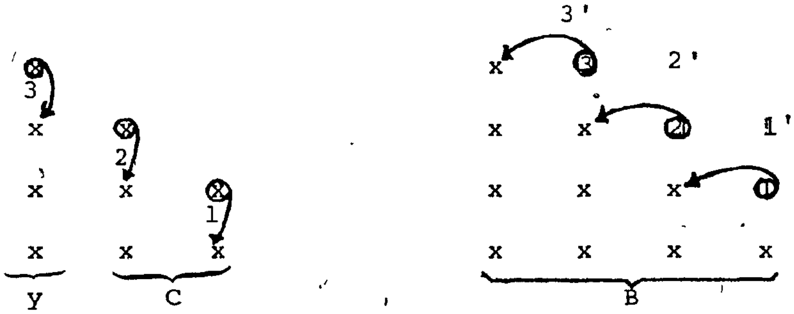
The transformations in (3.2) can be performed in two stages. In the first stage, we apply $n(n+1)/2$ rotations from the left to zero out the above main diagonal part of $[y \vdots C]$ with corresponding rotations automatically applied from the right to B, whenever necessary, to retain its form. For example with $m = 4$, $n = 2$, the initial step is



(3.9)

The rotations are ordered 1, 1', 2, 2'. The nonzero element $\boxed{i}$ introduced by ith rotation from left is immediately made zero by i' rotation from the right. Using this procedure, the above main diagonal part of [y,C] is reduced to zero , at the same time keeping the form of B the same throughout.

In the second stage of reduction each step has the same form, eliminating one diagonal of [y,C] matrix at a time and also maintianing the lower triangular form of B at the same time by applying right rotations. The first step of the second stage for the above case [m = 4, n = 2] will be as follows:



(3.10)

where the sequence of rotation is 1, 1', 2, 2', 3, 3'. So there will be m-n-1 steps for the second stage of the reduction and each step will contain (n+1) rotations to be applied to the left of [y,C,B] and (n+1) rotations to the right of B to maintain its form.

It is seen that after the first element of y is eliminated, there is no need to consider the first column of B any further, since the first column of B will no longer have any effect on the solution. Therefore the first column of B can be ignored completely once the first element of y is eliminated. Thus as

.../44

the reduction of the system continues, the size of the system also decreases. This will save computations in the general case and also save storage in sparse problems. Therefore, this algorithm is more efficient than the one given by Paige [17]. Paige [18] has also given the rounding error analysis of this algorithm and found it to be numerically stable.

The transformations given in (3.2) can also be achieved by using stabilized nonunitary transformations instead of Givens rotations from the left [18]. The transformation can be shown as follows:

$$\begin{bmatrix} 1 & -\alpha/\beta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 0 \\ \beta \end{bmatrix}, \text{ if } \beta \neq 0. \qquad (3.11)$$

To maintain numerical stability we would first permute the elements α and β if $\beta = 0$ or $|\alpha/\beta| > 1$. Thus we see that it needs only one multiplication to eliminate an element instead of 4 multiplications in the case of Givens plane rotations. Since (3.1) requires the minimization of $v^T v$ and nonunitary transformations do not preserve the 2-norm, it is better to use orthogonal transformations from the right. Thus we can produce the results of (3.2) by applying nonunitary transformations from the left and fast Givens rotations from the right [18].

.../45

3.3 Operation count.

It will be assumed for simplicity that C has full column rank and B is square. We will consider first that the reduction in (3.2) is achieved using 4 multiplication rotations only.

To compute the first stage, $n(n+1)/2$ rotations to the left of $[y,C,B]$ take about $2n^3$ operations and $n(n+1)/2$ rotations to the right of B take about $2mn^2 - \frac{2}{3}n^3$ operations. Hence the first stage of reductions takes about

$$2mn^2 + \frac{4}{3}n^3 \qquad (3.12)$$

operations.

The second stage of reduction contains $(m-n-1)$ steps. Each step consists of $(n+1)$ rotations to the left of $[y,C,B]$ and the corresponding $(n+1)$ rotations to the right of B . $(m-n-1)(n+1)$ left rotations take about $2m^2n-2n^3$ operations and $(m-n-1)(n+1)$ right rotations to B take $2m^2n-2mn^2$ operations. Therefore, total operations needed for second stage of the reduction is about

$$4m^2n - 2mn^2 - 2n^3 . \qquad (3.13)$$

Thus, the complete reduction takes about

$$4m^2n - \frac{2}{3}n^3 \qquad (3.14)$$

operations.

It has been shown that the matrix B can be reduced as the computation progresses. The total number of operations thus saved will be about $2n(m-n)^2$ operations. So the fast algorithm takes about

$$2m^2n + 4mn^2 - \frac{8}{3}n^3 \qquad (3.15)$$

operations to solve the generalized linear least squares problem.

The number of operations saved i.e. $2n(m-n)^2$ will be appreciable if $m >> n$. Thus we can say that the fast step by step method will be very efficient for a large general linear model.

If we use stabilized nonunitary transformations from the left instead of Givens rotations, then in the first stage, $n(n+1)/2$ transformations to the left of $[y,C,B]$ take about $n^3/2$ operations. Hence to accomplish the first stage requires about

$$2mn^2 - \frac{1}{6}n^3 \qquad (3.16)$$

operations.

In the second stage $(m-n-1)(n+1)$ transformations from the left take about $(m^2n-n^3)/2$ operations. Therefore, the second stage of the reduction takes about

$$\frac{5}{2}m^2n - 2mn^2 - \frac{n^3}{2} \qquad (3.17)$$

operations.

Thus, if nonunitary transformations from the left and Givens 4 multiplication rotations from the right are used in the reduction (3.2), total number of operations is about

$$\frac{5}{2} m^2 n - \frac{2}{3} n^3 . \qquad (3.18)$$

From (3.14) and (3.18) we find that using stabililized nonunitary transformations from the left and Givens 4 multiplication rotations from the right will always be faster than using 4 multiplication orthogonal rotations from the left and the right of the system.

The fast step by step algorithm for solving the generalized least squares problem (3.1) described in this chapter has been tested on an IBM/370 system. The procedure GLSQUARES, written in ALGOLW, has been presented in Appendix A. We have used 4 multiplication Givens rotations to reduce the system. The procedure works for general C and B. The outputs for the test problems are also given in Appendix A.

CHAPTER 4

MINIMUM 2-NORM SOLUTION OF A STRUCTURED UNDERDETERMINED SYSTEM.

4.1 Introduction.

In many generalized least squares problems (Paige [16], Theil [22] pp.294-299), it is necessary to find the minimum 2-norm solution of an underdetermined system of the form

$$y = F\ z \tag{4.1}$$

where F has the following structure

$$F = \begin{bmatrix} G_1 & L_1 & & & \\ G_2 & & L_2 & & \\ \cdot & & & \cdot & \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ G_n & & & & L_n \end{bmatrix} . \tag{4.2}$$

For each i, $i=1,2,...,n$, G_i is a matrix of order $m_i \times k$ and L_i is lower trapezoidal full column rank matrix of rank k_i. Elsewhere, all elements of F are zero. In Chapter 5 we will encounter a model where the efficient solution of an underdetermined system of the form (4.1) with (4.2) is required.

One way of finding a numerically stable solution to (4.1) is to find an orthogonal matrix Q such that

$$FQ = [\bar{L},\ 0] \tag{4.3}$$

where $\bar{L}$ is a full column rank matrix. Because of the particular structure of F, it is not worthwhile to construct Q or $\bar{L}$ ex-

.../49

plicitly since this will result in large storage requirements and unnecessary computations. In this Chapter we will present an apparently new and numerically stable algorithm, designed by C. Paige and the author, which requires very little storage and also avoids unnecessary computations. We will assume for simplicity in the description that the matrix F has full row rank and each matrix L_i for $i=1,2,...,n$ is lower triangular of order m_i. An operation count will also be given for this particular case. The procedure MINNORM given in the Appendix B also works when F is not a full row rank matrix and each L_i is a full column rank lower trapezoidal matrix. When F is not a full row rank matrix we can check the consistency of the system by checking the residue which will be small for compatible system.

4.2 Method of solution.

We can write (4.1) as follows

$$[G,L]\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = y \qquad (4.4)$$

where

$$\underset{mn\times k}{G} = \begin{bmatrix} G_1 \\ G_2 \\ . \\ . \\ G_n \end{bmatrix}, \quad \underset{mn\times mn}{L} = \begin{bmatrix} L_1 & & & \\ & L_2 & & \\ & & \ddots & \\ & & & L_n \end{bmatrix}, \quad \underset{(k+mn)\times 1}{z} = \begin{bmatrix} z_1 \\ \\ z_2 \end{bmatrix} \qquad (4.5)$$

Gill et al [6] presented an efficient algorithm for solving an underdetermined system of the form

$$[a,D]\begin{bmatrix}\nu\\ v\end{bmatrix} = b \qquad (4.6)$$

where D is a diagonal matrix, a,v and b are column vectors and ν is a scalar. The algorithm for solving the system (4.4) which will be presented here, can be considered as a generalization to block form of the algorithm presented by Gill et al [6].

We add to the bottom of the matrix in (4.4) a matrix

$$[I,0] \qquad (4.7)$$

where I is an identity matrix of rank k so that we will work with the system of the form

$$\begin{bmatrix}G & L\\ I & 0\end{bmatrix}. \qquad (4.8)$$

The reason for working with the system (4.8) is to form a part of the transformation matrix simultaneously as the reduction of the system progresses. Its purpose will be evident as we progress.

At first, we will form an orthogonal matrix $Q^{(1)}$ such that

$$m_1\{[\underbrace{G_1}_{k},\underbrace{L_1}_{m_1}]\ \underbrace{Q^{(1)}}_{(m_1+k)\times(m_1+k)} = [\underbrace{\bar{L}_1}_{m_1},\underbrace{0}_{k}],\ Q^{(1)} = \begin{array}{cc} \begin{bmatrix} Q^{(1)}_{11} & Q^{(1)}_{12}\\ Q^{(1)}_{21} & Q^{(1)}_{22}\end{bmatrix} & \begin{array}{l}\}k\\ \}m_1\end{array}\\ \begin{array}{cc}\underbrace{\quad}_{m_1} & \underbrace{\quad}_{k}\end{array} & \end{array} \qquad (4.9)$$

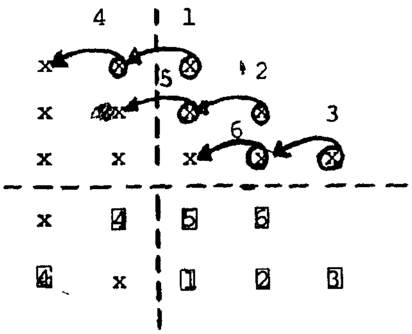
.../51

where $\bar{L}_1$ is a full column rank matrix. If we first multiply the system (4.8) by $\begin{bmatrix} Q^{(1)} & 0 \\ 0 & I \end{bmatrix}$ then (4.8) reduces to

$$\begin{bmatrix} \bar{L}_1 & & & \\ G_2Q_{11}^{(1)} & G_2Q_{12}^{(1)} & L_2 & \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \quad\cdot \\ G_nQ_{11}^{(1)} & G_nQ_{12}^{(1)} & & L_n \\ Q_{11}^{(1)} & Q_{12}^{(1)} & 0 & 0 \end{bmatrix} \quad . \tag{4.10}$$

If we use Givens plane rotations, then the reduction of $[G_1,L_1]$ to a full column rank matrix can be performed in such a way that $Q_{12}^{(1)}$ is lower triangular. This can be shown schematically, in the case of $m_1 = 3$, $k = 2$, $k_1 = 3$, as follows:

$$\left[\begin{array}{c|c} G_1 & L_1 \\ \hline I & 0 \end{array}\right] =$$



$$= \left[\begin{array}{c|c} \bar{L}_1 & 0 \\ \hline Q_{11}^{(1)} & Q_{12}^{(1)} \end{array}\right] = \left[\begin{array}{ccc|cc} x & & & & \\ x & x & & & \\ x & x & x & & \\ \hline x & x & x & x & \\ x & x & x & x & x \end{array}\right] \tag{4.11}$$

The rotations are ordered 1, 2, 3, 4, 5, 6, $\overset{i}{\otimes}$ indicating that the element has been made zero in the i th rotation. The nonzero element $\boxed{i}$ has been introduced by the i th rotation from the right.

Next, we form another orthogonal matrix $Q^{(2)}$ such that

$$m_2\{[\underbrace{G_2Q_{12}^{(1)}}_{k}, \underbrace{L_2}_{m_2}]Q^{(2)} = [\underbrace{\bar{L}_2}_{m_2}, \underbrace{0}_{k}], \; Q^{(2)} = \left[\begin{array}{cc} Q_{11}^{(2)} & Q_{12}^{(2)} \\ \\ \underbrace{Q_{21}^{(2)}}_{m_2} & \underbrace{Q_{22}^{(2)}}_{k} \end{array}\right]\begin{array}{l} \}k \\ \\ \}m_2 \end{array} \qquad (4.12)$$

where $\bar{L}_2$ is a full column rank matrix. Thus (4.10) can be transformed to

$$\left[\begin{array}{ccccc} \bar{L}_1 & & & & \\ G_2Q_{11}^{(1)} & \bar{L}_2 & & & \\ G_3Q_{11}^{(1)} & G_3Q_{12}^{(1)}Q_{11}^{(2)} & G_3Q_{12}^{(1)}Q_{12}^{(2)} & L_3 & \\ \vdots & \vdots & \vdots & \ddots & \\ G_nQ_{11}^{(1)} & G_nQ_{12}^{(1)}Q_{11}^{(2)} & G_nQ_{12}^{(1)}Q_{12}^{(2)} & & L_n \\ Q_{11}^{(1)} & Q_{12}^{(1)}Q_{11}^{(2)} & Q_{12}^{(1)}Q_{12}^{(2)} & 0 & 0 \end{array}\right] \qquad (4.13)$$

The rotations are so applied that $Q_{12}^{(1)}Q_{12}^{(2)}$ is lower triangular. Continuing like this, we can find an orthogonal matrix Q, which is the product of n orthogonal matrices, such that

.../53

$$\begin{bmatrix} G & L \\ I & 0 \end{bmatrix}^{Q} = \begin{bmatrix} \bar{L}_1 & & & & & \\ G_2S_1 & \bar{L}_2 & & & & \\ G_3S_1 & G_3S_2 & \bar{L}_3 & & & \\ . & . & . & . & & \\ G_nS_1 & G_nS_2 & G_nS_3 & . & \bar{L}_n & 0 \\ S_1 & S_2 & S_3 & . & S_n & Z_{n+1} \end{bmatrix} \begin{matrix} S_i = Q_{12}^{(1)} Q_{12}^{(2)} .. Q_{12}^{(i-1)} Q_{11}^{(i)} \\ , Z_{n+1} = Q_{12}^{(1)} Q_{12}^{(2)} \,.\,.\, Q_{12}^{(n)} \end{matrix}$$

$$= \begin{bmatrix} \bar{L} & 0 \\ Q_{11} & Q_{12} \end{bmatrix} \text{, say} \qquad (4.14)$$

Although this is a nonsparse lower triangular matrix, its special form allows us to form and use it with very little storage and computation. This will be described further in Section 4.3.

The system given in (4.4) can thus be transformed to the following

$$[\bar{L},\ 0]\, Q^T \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = y \ . \qquad (4.15)$$

Let

$$w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = Q^T \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

Then solving the compatible system

$$\bar{L}w_1 = y \tag{4.16}$$

in an efficient way for w_1 and setting $w_2 = 0$, the minimum 2-norm solution of (4.1) is given by

$$\hat{z} = \begin{bmatrix} \hat{z}_1 \\ \hat{z}_2 \end{bmatrix} = Q \begin{bmatrix} w_1 \\ 0 \end{bmatrix} . \tag{4.17}$$

But it can be seen in (4.14) that we know only a part of Q . However, (4.14) and (4.17) give

$$\hat{z}_1 = Q_{11}w_1 , \tag{4.18}$$

and this can be evaluated.

As we are concerned with the minimum 2-norm solution of (4.4) and since $\hat{z}_1$ is already evaluated from (4.18), then $\hat{z}_2$ is nothing but the solution of the compatible system

$$L\hat{z}_2 = y - G\hat{z}_1 . \tag{4.19}$$

This system is square if all L_i , i=1,2,...,n , are square. In any case (4.19) can easily be solved since the system is compatible and L is a block diagonal matrix.

4.3 Algorithm of the method.

We will present here the algorithm for evaluating $\hat{z}_1$ only. The remaining part of the minimum 2-norm solution can easily be obtained from (4.19) once $\hat{z}_1$ is known.

(4.16) can be written as

$$\begin{bmatrix} \bar{L}_1 & & & & \\ G_2S_1 & \bar{L}_2 & & & \\ G_3S_1 & G_3S_2 & \bar{L}_3 & & \\ \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \cdot & \\ G_nS_1 & G_nS_2 & G_nS_3 & \cdots & \bar{L}_n \end{bmatrix} \begin{bmatrix} w_{11} \\ w_{12} \\ w_{13} \\ \cdot \\ \cdot \\ \cdot \\ w_{1n} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix} \qquad (4.20)$$

where

$$S_i = Q_{12}^{(1)} \; Q_{12}^{(2)} \; \ldots \; Q_{12}^{(i-1)} \; Q_{11}^{(i)} \; ,$$

$$w_1^T = [w_{11}^T, w_{12}^T, \ldots, w_{1n}^T] \quad \text{and} \quad y^T = [\, y_1^T, y_2^T, \ldots, y_n^T] \; .$$

From (4.20) we can first find w_{11} by solving

$$\bar{L}_1 w_{11} = y_1 \; . \qquad (4.21)$$

Next we can obtain w_{12} by solving

$$\bar{L}_2 w_{12} = y_2 - G_2 d_1 \qquad (4.22)$$

for w_{12} where

$$d_1 = S_1 w_{11} \text{ , say .}$$

Thus w_{1i} can be evaluated by solving

$$\bar{L}_i w_{1i} = y_i - G_i d_{i-1} \tag{4.23}$$

for w_{1i} where

$$d_{i-1} = d_{i-2} + S_{i-1} w_{1,i-1} .$$

We are interested in finding d_n since from (4.18) we have

$$\begin{aligned} \hat{z}_1 &= S_1 w_{11} + S_2 w_{12} + \ldots + S_n w_{1n} \\ &= d_n . \end{aligned} \tag{4.24}$$

Therefore, we can describe the algorithm as follows:

```
i)   reduce $(G_1, L_1)$ to $(\bar{L}_1, 0)$, forming $S_1 := Q_{11}^{(1)}$, $Z_2 := Q_{12}^{(1)}$
ii)  solve $\bar{L}_1 w_{11} := y_1$
iii) form $d_1 := S_1 w_{11}$
iv)  for $i := 2, 3, \ldots, n$ do                                        (4.25)
         a)  reduce $(G_i Z_i, L_i)$ to $(\bar{L}_i, 0)$
             forming $S_i := Z_i Q_{11}^{(i)}$, $Z_{i+1} := Z_i Q_{12}^{(i)}$
         b)  solve $\bar{L}_i w_{1i} := y_i - G_i d_{i-1}$
         c)  form $d_i := d_{i-1} + S_i w_{1i}$ .
```

From (4.24) we find that d_n is our vector $\hat{z}_1$. Each system in steps (ii) and (iv-b) of the algorithm must be a compatible lower trapezoidal system and can easily be solved (In most cases, all L_i will be lower triangular).

Thus, we see that the algorithm for evaluating $\hat{z}_1$ is very efficient. The method is basically sequential and at no stage need we store previous $\bar{L}_i$ or S_i, and Z_i is overwritten by Z_{i+1}. This special form leads to tremendous savings with respect to computation time and storage requirements.

A computer program for obtaining $\hat{z}_1$ written in ALGOLW, together with the output for test problems run on IBM 370, are given in Appendix B.

We see that when $\hat{z}_1$ has been found, $\hat{z}_2$ can be found by keeping L and G and solving the compatible system (4.19).

4.4 Operation count.

We now give an operation count for the algorithm given in (4.25) to evaluate $\hat{z}_1$. We will assume, for simplicity, that all L_i, $i=1,2,...,n$, are nonsingular lower triangular matrices of order m. We will also assume that the matrix F in (4.1) has full row rank. We will apply 4 multiplication Givens rotations for the reduction of the system.

We examine the number of operations required in step (iv) of the algorithm given in (4.25). To form G_iZ_i where Z_i is lower triangular takes $\frac{1}{2}mk^2$ operations. Reduction of $[G_iZ_i, L_i]$ to $[\bar{L}_i, 0]$ and at the same time forming S_i and Z_{i+1} as described in (4.11) takes about $2m^2k + 2mk^2$ operations. Since the other operations are relatively smaller, we can say that the total operations necessary to implement the algorithm

.../58

given in (4.25) is about

$$2m^2nk + \frac{5}{2} mnk^2 \quad . \tag{4.26}$$

Once $\hat{z}_1$ is known, $\hat{z}_2$ can be evaluated by solving (4.19). Total operations required to solve (4.19) is about $\frac{1}{2} m^2 n$ operations. Therefore, to solve the system (4.1) completely takes about

$$m^2n(2k + \frac{1}{2}) + \frac{5}{2} mnk^2 \tag{4.27}$$

operations.

If we solve (4.1) by reducing it to the form (4.3) by forming Q and $\bar{L}$ then the total number of operations necessary is of the order of

$$m^2n^2k \quad . \tag{4.28}$$

We see that if fast 2-multiplication Givens rotations are applied for implementing algorithm (4.25), then the algorithm presented here is about n times faster than the one that reduces (4.1) to the form (4.3). Moreover, for our algorithm, practically no extra storage is required besides storing G_i and L_i, $i=1,2,\ldots,n$.

.../59

CHAPTER 5

A REPEATABLE EXPERIMENT WITH A GENERAL LINEAR MODEL.

5.1 Introduction.

In Section 2.4.1 we have seen that if in a controlled experiment, each set of observations is taken with a different set of measuring instruments then it is possible to have different variance-covariance matrices for the noise terms. Let W_i be the variance-covariance matrix of the noise vector for the ith observation $\bar{y}_i$. Let $W_i = \bar{B}_i\bar{B}_i^T$ be the Cholesky decomposition where $\bar{B}_i$ is lower trapezoidal with full column rank. Then for t sets of observations, we can write the linear model as follows:

$$y = Cx + Bv; \; E(v) = 0, \; E(vv^T) = \sigma^2 I \qquad (5.1)$$

where

$$y = \begin{bmatrix} \bar{y}_1 \\ \bar{y}_2 \\ \cdot \\ \cdot \\ \cdot \\ \bar{y}_t \end{bmatrix}, \; C = \begin{bmatrix} \bar{C} \\ \bar{C} \\ \cdot \\ \cdot \\ \cdot \\ \bar{C} \end{bmatrix}, \; B = \begin{bmatrix} \bar{B}_1 & & & \\ & \bar{B}_2 & & \\ & & \ddots & \\ & & & \bar{B}_t \end{bmatrix} \qquad (5.2)$$

where $\bar{y}_i$ is an m-vector, $\bar{C}$ is an $m\times n$ matrix and $\bar{B}_i$ is a full column rank matrix of dimension $m\times k_i$.

We will develop a numerically stable algorithm to obtain the generalized least squares estimate $\hat{x}$ for this linear model. We will also present a very fast algorithm to solve this problem

.../60

when $\bar{B}_i = \bar{B}$ for all i. This algorithm is developed jointly with C. Paige.

We can formulate our problem as

$$\min_{v,x} v^T v \quad \text{subject to} \quad y = Cx + Bv \tag{5.3}$$

with y,C,B given in (5.2). The vector x solving (5.3) is the required estimate $\hat{x}$.

5.2 Method of solution.

To begin with, we first find an orthogonal matrix $\bar{Q}$ such that

$$\bar{Q}^T\bar{C} = \begin{bmatrix} 0 \\ \\ R^T \end{bmatrix} \tag{5.4}$$

where R^T is a full row rank matrix. We also find an orthogonal matrix P_i such that

$$\bar{Q}^T\bar{B}_iP_i = \bar{B}_i' = \begin{bmatrix} L_i & \\ F_i & N_i \end{bmatrix}, \ i=1,2,\ldots,t \tag{5.5}$$

where the forms of $\bar{B}_i$ and $\bar{B}_i'$ are the same. The matrix $\bar{B}_i'$ is partitioned so that the number of rows in $[F_i, N_i]$ is the same as that in R^T in (5.4). Then the constraints in (5.3) transform to the following:

.../61

$$Q^T y = \begin{bmatrix} 0 \\ R^T \\ 0 \\ R^T \\ \cdot \\ \cdot \\ 0 \\ R^T \end{bmatrix} x + \begin{bmatrix} L_1 & & & & & \\ F_1 & N_1 & & & & \\ & & L_2 & & & \\ & & F_2 & N_2 & & \\ & & & & \ddots & \\ & & & & L_t & \\ & & & & F_t & N_t \end{bmatrix} P^T v \qquad (5.6)$$

where

$$Q^T = \begin{bmatrix} \bar{Q}^T & & & \\ & \bar{Q}^T & & \\ & & \ddots & \\ & & & \bar{Q}^T \end{bmatrix}, \quad P = \begin{bmatrix} P_1 & & & \\ & P_2 & & \\ & & \ddots & \\ & & & P_t \end{bmatrix} .$$

Let

$$y' = Q^T y = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_t \end{bmatrix} = \begin{bmatrix} y_{11} \\ y_{12} \\ y_{21} \\ y_{22} \\ \cdot \\ \cdot \\ y_{t1} \\ y_{t2} \end{bmatrix} \text{ and } v' = P^T v = \begin{bmatrix} v_{11} \\ v_{12} \\ v_{21} \\ v_{22} \\ \cdot \\ v_{t1} \\ v_{t2} \end{bmatrix},$$

$$y_i = \begin{bmatrix} y_{i1} \\ \\ y_{i2} \end{bmatrix}, \quad v_i = \begin{bmatrix} v_{i1} \\ \\ v_{i2} \end{bmatrix} \qquad (5.7)$$

We can split (5.6) into the following two parts after rearranging its equations:

.../62

$$\begin{bmatrix} y_{12} \\ y_{22} \\ \cdot \\ \cdot \\ y_{t2} \end{bmatrix} = \begin{bmatrix} R^T \\ R^T \\ \cdot \\ \cdot \\ R^T \end{bmatrix} x + \begin{bmatrix} F_1 \; N_1 & & & \\ & F_2 \; N_2 & & \\ & & \ddots & \\ & & & F_t \; N_t \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{12} \\ v_{21} \\ v_{22} \\ \cdot \\ v_{t1} \\ v_{t2} \end{bmatrix} \qquad (5.8)$$

and

$$\begin{bmatrix} y_{11} \\ y_{21} \\ \cdot \\ \cdot \\ y_{t1} \end{bmatrix} = \begin{bmatrix} L_1 \; 0 & & & \\ & L_2 \; 0 & & \\ & & \ddots & \\ & & & L_t \; 0 \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{12} \\ v_{21} \\ v_{22} \\ \cdot \\ \cdot \\ v_{t1} \\ v_{t2} \end{bmatrix} . \qquad (5.9)$$

The system (5.9) can be solved easily for $[\, v_{11}^T, v_{21}^T, \ldots, v_{t1}^T]^T$ and after substituting back in (5.8) we have

$$\begin{bmatrix} y_{12} - F_1 v_{11} \\ y_{22} - F_2 v_{21} \\ \cdot \\ \cdot \\ \cdot \\ y_{t2} - F_t v_{t1} \end{bmatrix} = \begin{bmatrix} R^T \\ R^T \\ \cdot \\ \cdot \\ \cdot \\ R^T \end{bmatrix} x + \begin{bmatrix} N_1 & & & \\ & N_2 & & \\ & & \ddots & \\ & & & N_t \end{bmatrix} \begin{bmatrix} v_{12} \\ v_{22} \\ \cdot \\ \cdot \\ \cdot \\ v_{t2} \end{bmatrix} . \qquad (5.10)$$

If any L_i in (5.9) is nonsquare, we can find an orthogonal matrix $\tilde{Q}_i$ such that

$$\tilde{Q}_i^T L_i = \begin{bmatrix} 0 \\ \\ \bar{L}_i \end{bmatrix} \qquad (5.11)$$

.../63

where $\bar{L}_i$ is lower triangular. In this case we can check the consistency of the model by observing the residual. For a consistent model, the computed residual must be small.

Let us now multiply both sides of (5.10) by the non-singular matrix

$$\begin{bmatrix} I & & & & \\ -I & I & & & \\ \cdot & & \cdot & & \\ \cdot & & & \cdot & \\ \cdot & & & & \cdot \\ -I & & & & I \end{bmatrix} \tag{5.12}$$

so that (5.10) transforms to

$$\begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_t \end{bmatrix} = \begin{bmatrix} R^T \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} x + \begin{bmatrix} N_1 & & & & \\ -N_1 & N_2 & & & \\ \cdot & & \cdot & & \\ \cdot & & & \cdot & \\ \cdot & & & & \cdot \\ -N_1 & & & & N_t \end{bmatrix} \begin{bmatrix} v_{12} \\ v_{22} \\ \cdot \\ \cdot \\ \cdot \\ v_{t2} \end{bmatrix} \tag{5.13}$$

We can find $\hat{x}$ by solving

$$b_1 = R^T\hat{x} + N_1 v_{12} \tag{5.14}$$

once v_{12} is known. Therefore, we need to find the minimum 2-norm solution of the following underdetermined system

$$\begin{bmatrix} b_2 \\ b_3 \\ \cdot \\ \cdot \\ \cdot \\ b_t \end{bmatrix} = \begin{bmatrix} -N_1 & N_2 & & & \\ -N_1 & & N_3 & & \\ \cdot & & & \cdot & \\ \cdot & & & & \cdot \\ \cdot & & & & \\ -N_1 & & & & N_t \end{bmatrix} \begin{bmatrix} v_{12} \\ v_{22} \\ \cdot \\ \cdot \\ \cdot \\ v_{t2} \end{bmatrix} \qquad (5.15)$$

or

$$\bar{b} = \bar{N}\bar{v} \text{ , say .} \qquad (5.16)$$

The method of solving an underdetermined system of the form (5.16) has already been described in Chapter 4. Once $\bar{v}$ is known, we can solve (5.14) for $\hat{x}$ which will be the least squares estimate of the model (5.1). We see that in order to estimate $\hat{x}$ from (5.14) we need to know only $\hat{v}_{12}$. We know that the algorithm presented in Chapter 4 is very efficient in obtaining $\hat{v}_{12}$ only.

We can solve the problem (5.3) quickly if the variance-covariance matrices of the noise vectors for all the t different experiments are the same. If we follow the same methods as described before, (5.5) will be of the form

$$\bar{Q}^T\bar{B}\bar{P} = \bar{B}' = \begin{bmatrix} L & \\ F & N \end{bmatrix} . \qquad (5.17)$$

$\hat{x}$ can then be found by solving

$$b_1 = R^T\hat{x} + Nv_{12} \qquad (5.18)$$

.../65

once v_{12} is known, and the constraints on $\bar{v}$ are now given by

$$\bar{b} = \bar{N}\bar{v} \tag{5.19}$$

where

$$\bar{N} = \begin{bmatrix} -N & N & & & \\ -N & & N & & \\ \cdot & & & \cdot & \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ -N & & & & N \end{bmatrix}, \quad \bar{b} = \begin{bmatrix} b_2 \\ b_3 \\ \cdot \\ \cdot \\ \cdot \\ b_t \end{bmatrix}, \quad \bar{v} = \begin{bmatrix} v_{12} \\ v_{22} \\ \cdot \\ \cdot \\ \cdot \\ v_{t2} \end{bmatrix}. \tag{5.20}$$

We can apply a technique similar to Givens plane rotations, which we will call multidimensional rotations, to transform $\tilde{N}$ to a lower triangular form. Multidimensional rotation matrix is defined as

$$\begin{bmatrix} -\alpha I & \beta I \\ \beta I & \alpha I \end{bmatrix} . \tag{5.21}$$

The choice of α and β to perform the reduction

$$[\, -\delta N,\ N \,] \begin{bmatrix} -\alpha I & \beta I \\ \beta I & \alpha I \end{bmatrix} = [\, \gamma N,\ 0 \,] ,\ \delta > 0 \tag{5.22}$$

is given by

$$\alpha = \frac{\delta}{\sqrt{1+\delta^2}} \quad \text{and} \quad \beta = \frac{1}{\sqrt{1+\delta^2}} \tag{5.23}$$

.../66

and

$$\gamma = \sqrt{1 + \delta^2} \ . \tag{5.24}$$

We can write the system (5.19) as follows

$$[N_1, N_2] \begin{bmatrix} v_{12} \\ . \\ v' \end{bmatrix} = \bar{b} \tag{5.25}$$

where

$$N_1 = \begin{bmatrix} -N \\ -N \\ . \\ . \\ -N \end{bmatrix}, \ N_2 = \begin{bmatrix} N & & & \\ & N & & \\ & & . & \\ & & & . \\ & & & & N \end{bmatrix}, \ v' = \begin{bmatrix} v_{22} \\ v_{32} \\ . \\ . \\ v_{t2} \end{bmatrix}. \tag{5.26}$$

As we have done in Chapter 4, we will consider the system

$$\begin{bmatrix} N_1 & N_2 \\ I & 0 \end{bmatrix} \tag{5.27}$$

where I is an identity matrix. This may help motivate the algorithm we will give at the end of section 5.3.

We will reduce the system $[N_1, N_2]$ using multidimensional rotations. We can find an orthogonal matrix $P^{(1)}$ of the form

$$P^{(1)} = \begin{bmatrix} -\alpha_1 I & \beta_1 I \\ \beta_1 I & \alpha_1 I \end{bmatrix} \tag{5.28}$$

such that

$$[-N,\ N]\,P^{(1)} = [\gamma_1 N, 0] \tag{5.29}$$

where α_1, β_1, γ_1 are given by α, β, γ in (5.23) and (5.24) with $\delta = 1$. Then (5.27) can be transformed to

$$\begin{bmatrix} \gamma_1 N & 0 & & & & \\ \alpha_1 N & -\beta_1 N & N & & & \\ \alpha_1 N & -\beta_1 N & & N & & \\ \cdot & & & & \cdot & \\ \cdot & & & & \cdot & \\ \cdot & & & & & \cdot \\ \alpha_1 N & -\beta_1 N & & & & N \\ \alpha_1 I & \beta_1 I & 0 & 0 & \ldots & 0 \end{bmatrix} \tag{5.30}$$

Next our $\delta = \beta_1$. Applying the same technique, (5.30) can further be reduced to

$$\begin{bmatrix} \gamma_1 N & & & & & \\ \alpha_1 N & \gamma_2 N & 0 & & & \\ \alpha_1 N & \beta_1\alpha_2 N & -\beta_1\beta_2 N & N & & \\ \cdot & \cdot & \cdot & & \cdot & \\ \cdot & \cdot & \cdot & & & \cdot \\ \cdot & \cdot & \cdot & & & \\ \alpha_1 N & \beta_1\alpha_2 N & -\beta_1\beta_2 N & & & N \\ \alpha_1 I & \beta_1\alpha_2 I & \beta_1\beta_2 I & 0 & \ldots & 0 \end{bmatrix} \tag{5.31}$$

where our next $\delta = \beta_1 \beta_2$. Continuing like this, at the end, we will have

$$\begin{bmatrix} \gamma_1 N & & & & & \\ \alpha_1 N & \gamma_2 N & & & & \\ \alpha_1 N & \beta_1 \alpha_2 N & \gamma_3 N & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ \alpha_1 N & \beta_1 \alpha_2 N & \beta_1 \beta_2 \alpha_3 N & \cdots & \gamma_{t-1} N & 0 \\ S_1 & S_2 & S_3 & \cdots & S_{t-1} & Z_t \end{bmatrix} , \quad \begin{matrix} S_i = \beta_1 \beta_2 .. \beta_{i-1} \alpha_i I , \\ Z_t = \beta_1 \beta_2 .. \beta_{t-1} I \end{matrix} \tag{5.32}$$

$$= \begin{bmatrix} \bar{L} & 0 \\ P_{11} & P_{12} \end{bmatrix} , \text{ say} . \tag{5.33}$$

Thus there exist an orthogonal matrix $\tilde{P}$ such that the underdetermined system (5.19) can be transformed to

$$[\bar{L}, 0] \tilde{P}^T \bar{v} = \bar{b} , \quad \tilde{P} = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \tag{5.34}$$

where $\tilde{P}$ is the product of (t-1) multidimensional rotation matrices.

Now

$$\tilde{P}^T \bar{v} = \tilde{P}^T \begin{bmatrix} v_{12} \\ v' \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} , \text{ say.} \tag{5.35}$$

Then solving the compatible system

$$\bar{L}w_1 = \bar{b} \qquad (5.36)$$

in an efficient way for w_1 and setting $w_2 = 0$, the minimum 2-norm solution of (5.25) is given by

$$\begin{bmatrix} \hat{v}_{12} \\ \hat{v}' \end{bmatrix} = \tilde{P} \begin{bmatrix} w_1 \\ 0 \end{bmatrix} = \begin{bmatrix} P_{11}w_1 \\ P_{21}w_1 \end{bmatrix} . \qquad (5.37)$$

Thus, we can evaluate $\hat{v}_{12}$ easily as P_{11} is known. Since P_{21} is not known, we cannot evaluate $\hat{v}'$ from (5.37). As we are concerned with the minimum 2-norm solution of (5.25) and since $\hat{v}_{12}$ is already known, we can obtain $\hat{v}'$, if it is required, by solving the following compatible block diagonal system [from (5.25)]

$$N_2\hat{v}' = \bar{b} - N_1\hat{v}_{12} . \qquad (5.38)$$

(5.38) can be solved very efficiently since N_2 and N_1 have special forms (5.26).

5.3 Algorithm of the method.

We will present here the algorithm for the case when the variance-covariance matrices of the noise terms are all the same i.e. when multidimensional rotations are used to reduce the system.

Our aim will be to evaluate $\hat{v}_{12}$ first and then the rest of the solution vector can be obtained by solving (5.38). We will follow the same approach as we have done in the Chapter 4. We will modify it to take advantage of the special forms of N_1 and N_2 given in (5.26).

We can write (5.36) as follows

$$\bar{L}w_1 = \begin{bmatrix} \gamma_1 N & & & & \\ S_1 N & \gamma_2 N & & & \\ S_1 N & S_2 N & \gamma_3 N & & \\ \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ S_1 N & S_2 N & S_3 N & \cdots & \gamma_{t-1} N \end{bmatrix} \begin{bmatrix} w_{11} \\ w_{12} \\ w_{13} \\ \cdot \\ \cdot \\ \cdot \\ w_{1(t-1)} \end{bmatrix} = \begin{bmatrix} b_2 \\ b_3 \\ b_4 \\ \cdot \\ \cdot \\ \cdot \\ b_t \end{bmatrix} \qquad (5.39)$$

where

$$S_i = \beta_1 \beta_2 \ . \ . \ . \ \beta_{i-1} \alpha_i I \ .$$

From (5.39) we first find that

$$Nw_{11} = \frac{1}{\gamma_1} b_2 \ . \qquad (5.40)$$

Note at this point that we do not intend to solve for w_{11}.

Next Nw_{12} can be obtained as

$$Nw_{12} = \frac{1}{\gamma_2} (b_3 - d_1) \qquad (5.41)$$

.../71

where

$$d_1 = S_1 Nw_{11}$$

and again we do not solve.

Hence for i th block of equations, we have

$$Nw_{1i} = \frac{1}{\gamma_i} (b_{i+1} - d_{i-1}) \qquad (5.42)$$

where

$$d_{i-1} = d_{i-2} + S_{i-1} Nw_{1,i-1} .$$

The purpose of evaluating Nw_{1i}, not w_{1i} for $i=1,2,\ldots,t-1$, is that we need to know only Nw_{1i} in order to evaluate d_i. We are interested in evaluating d_i only because from (5.37) we have

$$\begin{aligned} N\hat{v}_{12} &= NP_{11}w \\ &= S_1 Nw_{11} + S_2 Nw_{12} + .. + S_{t-1} Nw_{1,t-1} \qquad (5.43) \\ &= d_{t-1} \end{aligned}$$

which could be used in (5.18).

Thus we describe the algorithm for estimating $\hat{x}$ as follows for the case when the variance-covariance matrices of the noise terms for different experiments are all the same.

.../72

```
i)    reduce $[\bar{y}_1,\bar{y}_2,\ldots,\bar{y}_t,\bar{C},\bar{B}]$ to $\begin{bmatrix} y_{11} & y_{21} & \cdots & y_{t1} & 0 & L & 0 \\ y_{12} & y_{22} & \cdots & y_{t2} & R^T & F & N \end{bmatrix}$
ii)   solve $Lv_{11} := y_{11}$ for $v_{11}$
iii)  form $b_1 := y_{12} - F \cdot v_{11}$
iv)   for $i := 2,3,\ldots,t$ do
          a)   solve $Lv_{i1} := y_{i1}$ for $v_{i1}$
          b)   form $f_i := y_{i2} - F\ v_{i1}$
          c)   form $b_i := -b_1 + f_i$
v)    initialize $\delta := 1$
vi)   evaluate $\alpha_1, \beta_1, \gamma_1$ as in (5.23) and (5.24)
vii)  form $g_1 := \frac{1}{\gamma_1} b_2$
viii) form $d_1 := \alpha_1 g_1$, $\mu_2 := \beta_1$
ix)   for $i := 2,3,\ldots,t-1$ do
          a)   set $\delta := \mu_i$
          b)   evaluate $\alpha_i, \beta_i, \gamma_i$ as in (5.23) and (5.24)
          c)   form $g_i := \frac{1}{\gamma_i}(b_{i+1} - d_{i-1})$
          d)   form $d_i := d_{i-1} + \mu_i \alpha_i g_i$ , $\mu_{i+1} := \mu_i \beta_i$
x)    solve $R^T\hat{x} := b_1 - d_{t-1}$ for $\hat{x}$ .
```

.../73

If one is interested in obtaining $\hat{v}$ explicitly then $\hat{v}_{12}$ can be obtained by solving the compatible system (5.43) for $\hat{v}_{12}$. The remainder of the vector can be obtained by solving (5.38) where

$$N_1\hat{v}_{12} = \begin{bmatrix} -d_{t-1} \\ -d_{t-1} \\ \cdot \\ \cdot \\ -d_{t-1} \end{bmatrix} . \qquad (5.44)$$

5.4 Operation count.

We will first give an operation count required to estimate $\hat{x}$ for the model (5.1) when the variance-covariance matrices of the noise terms are all the same. We will assume that $\bar{C}$ is full column rank matrix of dimension $m x n$. We will also assume that $\bar{B}$ is lower triangular matrix of order m and also it is nonsingular. We will use 4 multiplication Givens rotations in reducing the system.

Step (i) of the algorithm takes about

$$4m^2n - \frac{2}{3}n^3 + 4mnt - 2n^2t \qquad (5.45)$$

operations.

Steps (ii), (iii) and (iv) take about

$$\frac{1}{2}m^2t - \frac{1}{2}n^2t \qquad (5.46)$$

operations.

Since the other operations of the algorithm are relatively smaller, the total operations needed to find $\hat{x}$ is approximately given by

$$4m^2n + \frac{1}{2} m^2t + 4mnt - \frac{5}{2} n^2t - \frac{2}{3} n^3 \quad . \qquad (5.47)$$

We will now count the number of operations necessary to compute $\hat{x}$ for the case when the variance-covariance matrices of the noise terms are different. We will assume that each matrix B_i in (5.2) is lower triangular and nonsingular of order m .

To compute (5.4) and (5.5) for $i=1,2,\ldots,t$, the total operations required is of the order of

$$4m^2nt - 2mn^2t \quad . \qquad ([illegible]48)$$

Since the other operations are relatively smaller, we can say that the total number of operations required to reduce the system to the form (5.13) is given by (5.48).

From Chapter 4 we find that the total number of operations necessary to evaluate v_{12} from (5.15) is about

$$2m^2nt + \frac{5}{2} mn^2t \quad . \qquad (5.49)$$

Therefore, the number of operations necessary to find $\hat{x}$ of the model (5.1) when the variance-covariance matrices of the noise vectors are different for different experiments is of the order of

$$6m^2nt + \frac{1}{2} mn^2t \quad . \qquad (5.50)$$

CHAPTER 6

GROUPING OF EQUATIONS.

6.1 Introduction.

We have seen in Section 2.4.2 that a set of different ordinary linear models whose noise terms are correlated can give rise to a general linear model of the form

$$y = Cx + u; \; E(u) = 0, \; E(uu^T) = W. \qquad (6.1)$$

The forms of y,C,W are

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix}, \quad C = \begin{bmatrix} C_1 & & & \\ & C_2 & & \\ & & \ddots & \\ & & & C_n \end{bmatrix}, \quad W = \begin{bmatrix} \alpha_{11}I & \alpha_{12}I & \dots & \alpha_{1n}I \\ \alpha_{21}I & \alpha_{22}I & \dots & \alpha_{2n}I \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \alpha_{n1}I & \alpha_{n2}I & \dots & \alpha_{nn}I \end{bmatrix} \qquad (6.2)$$

where y_i is an m-vector, C_i is an $m \times k_i$ matrix and W, the variance-covariance matrix of the random noise vector u, is of dimension mn × mn.

W, given in (6.2), can be written as

$$W = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \dots & \alpha_{1n} \\ \alpha_{21} & \alpha_{22} & \dots & \alpha_{2n} \\ \vdots & \vdots & & \vdots \\ \alpha_{n1} & \alpha_{n2} & \dots & \alpha_{nn} \end{bmatrix} \otimes I$$

$$= S \otimes I, \text{ say,} \qquad (6.3)$$

where $\otimes$ is the Kronecker product and I is a unit matrix of order m.

Let $S = LL^T$ be the Cholesky decomposition of S where

$$L = \begin{bmatrix} \sigma_{11} & & & \\ \sigma_{21} & \sigma_{22} & & \\ \vdots & \vdots & \ddots & \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_{nn} \end{bmatrix} \qquad (6.4)$$

Then (6.3) can be written as

$$\begin{aligned} W &= (L \otimes I)\,(L^T \otimes I) \\ &= BB^T \text{ , say} \end{aligned} \qquad (6.5)$$

where

$$B = \begin{bmatrix} \sigma_{11}I & & & \\ \sigma_{21}I & \sigma_{22}I & & \\ \vdots & \vdots & \ddots & \\ \sigma_{n1}I & \sigma_{n2}I & \cdots & \sigma_{nn}I \end{bmatrix} \qquad (6.6)$$

Therefore, the problem of least squares estimation of the general linear model given in (6.1) can be reformulated as

$$\underset{v,x}{\text{minimize}} \quad v^T v \quad \text{subject to } y = Cx + Bv \ . \qquad (6.7)$$

B can be nonsquare when the variance-covariance matrix W is singular. We will present here an algorithm which will work for linearly dependent columns of C_i and also of nonsquare B. This algorithm is developed jointly by C. Paige and the author.

6.2 Method of solution.

We can find an orthogonal matrix $Q^{(i)}$ such that

$$Q^{(i)^T} C_i = \begin{bmatrix} 0 \\ R_i^T \end{bmatrix} , \ Q^{(i)^T} = \begin{bmatrix} Q_1^{(i)^T} \\ Q_2^{(i)^T} \end{bmatrix} , \quad i = 1,2,...,n \qquad (6.8)$$

.../77

where R_i^T is a full row rank matrix and the orthogonal matrix $Q^{(i)^T}$ is partitioned so that the number of rows in R_i^T and $Q_2^{(i)^T}$ are the same.

The constraints given in (6.7) can now be transformed to

$$\begin{bmatrix} \bar{y}_1 \\ \bar{y}_2 \\ \cdot \\ \cdot \\ \cdot \\ \bar{y}_n \end{bmatrix} = \begin{bmatrix} \begin{matrix} 0 \\ R_1^T \end{matrix} & & & \\ & \begin{matrix} 0 \\ R_2^T \end{matrix} & & \\ & & \ddots & \\ & & & \begin{matrix} 0 \\ R_n^T \end{matrix} \end{bmatrix} x + \begin{bmatrix} \sigma_{11}Q^{(1)^T} & & & \\ \sigma_{21}Q^{(2)^T} & \sigma_{22}Q^{(2)^T} & & \\ \vdots & \vdots & \ddots & \\ \sigma_{n1}Q^{(n)^T} & \sigma_{n2}Q^{(n)^T} & \cdots & \sigma_{nn}Q^{(n)^T} \end{bmatrix} v \qquad (6.9)$$

where $\bar{y}_i = Q^{(i)^T} y_i$.

Rearranging the equations, we can split (6.9) into the following two parts

$$\begin{bmatrix} y_{12} \\ y_{22} \\ \cdot \\ \cdot \\ \cdot \\ y_{n2} \end{bmatrix} = \begin{bmatrix} R_1^T & & & \\ & R_2^T & & \\ & & \ddots & \\ & & & R_n^T \end{bmatrix} x + \begin{bmatrix} \sigma_{11}Q_2^{(1)^T} & & & \\ \sigma_{21}Q_2^{(2)^T} & \sigma_{22}Q_2^{(2)^T} & & \\ \vdots & \vdots & \ddots & \\ \sigma_{n1}Q_2^{(n)^T} & \sigma_{n2}Q_2^{(n)^T} & \cdots & \sigma_{nn}Q_2^{(n)^T} \end{bmatrix} v \qquad (6.10)$$

and

$$\begin{bmatrix} y_{11} \\ y_{21} \\ \cdot \\ \cdot \\ y_{n1} \end{bmatrix} = \begin{bmatrix} \sigma_{11}Q_1^{(1)^T} & & & \\ \sigma_{21}Q_1^{(2)^T} & \sigma_{22}Q_1^{(2)^T} & & \\ \vdots & \vdots & \ddots & \\ \sigma_{n1}Q_1^{(n)^T} & \sigma_{n2}Q_1^{(n)^T} & \cdots & \sigma_{nn}Q_1^{(n)^T} \end{bmatrix} v \qquad (6.11)$$

where

$$\bar{y}_i = \begin{bmatrix} y_{i1} \\ \\ y_{i2} \end{bmatrix} , \quad i=1,2,\ldots,n \ .$$

Thus we can estimate $\hat{x}$ by solving (6.10) for x once v is known. The problem (6.7) now reduces to

$$\min v^T v \ \text{ with respect to } \ v \ \text{ subject to (6.11)} \qquad (6.12)$$

The problem given in (6.12) is nothing but finding the minimum 2-norm solution of the underdetermined system (6.11) which can be written as

$$\bar{y} = Gv \ . \qquad (6.13)$$

The efficiency of the method of solving (6.7) depends largely on how efficiently we can solve (6.13). One way to solve (6.13) is by reducing G to a triangular form by applying orthogonal transformation P from the right. For large n and m, G can be very large and therefore, it is expensive to construct P and solve (6.13).

According to Peters and Wilkinson [19] the minimum 2-norm solution of (6.13) is given by

$$v^* = G^T(GG^T)^{-1}\bar{y} \ . \qquad (6.14)$$

So we could first form GG^T which is a symmetric positive definite matrix and then solve

$$GG^Tz = \bar{y} \qquad (6.15)$$

for z . The vector v can then be evaluated from

$$v = G^Tz \ . \qquad (6.16)$$

The system (6.15) can best be solved by using Cholesky decomposition of GG^T . Let

$$GG^T = LL^T \ , \ L \ \text{lower triangular} \ ,$$

be the Cholesky factorization. Then solving

$$Ly = \bar{y}$$

for y , we obtain z in (6.15) by solving

$$L^Tz = y \ .$$

There may be a faster but less obvious way, which we have not found, to solve (6.13) by reducing G to a lower triangular matrix of the form $[\bar{L},0]$ by applying orthogonal transformations P from the right without forming $\bar{L}$ or P explicitly.

Once v is known, we can estimate $\hat{x}$ by solving (6.10) for x .

6.3 Operation count.

For simplicity, we will assume that all the matrices C_i , $i=1,2,\ldots,n$, are of the same dimension $m\times k$ and have full column rank. In this case, we will use Householder transformation matrices for the reduction of the system.

The total number of operations required to reduce C_i to $\begin{bmatrix} 0 \\ \\ R_i^T \end{bmatrix}$ and form $Q^{(i)}$ at the same time is

$$2m^2k - \frac{1}{3}k^3 . \qquad (6.17)$$

Hence to reduce all the C_i to lower triangular form and to form the $Q^{(i)}$ require

$$2nm^2k - \frac{1}{3}nk^3 \qquad (6.18)$$

operations.

To form GG^T given in (6.15) takes about

$$\frac{1}{2}n^2m(m-k)^2 \qquad (6.19)$$

operations and the Cholesky decomposition of GG^T takes about

$$\frac{1}{6}n^3(m-k)^3 \qquad (6.20)$$

operations.

Once v is known, the total number of operations required to evaluate x from (6.10) is of the order of

$$\frac{1}{2}n^2mk . \qquad (6.21)$$

Hence the total number of operations required to obtain the solution of the problem given in (6.7) is about

.../81

$$2nm^2k - \frac{1}{3}nk^3 + \frac{1}{2}n^2m(m-k)^2 + \frac{1}{6}n^3(m-k)^3 + \frac{1}{2}n^2mk. \qquad (6.22)$$

We have seen in Section 2.4.2 that Zellner's method [23] of solving the problem (6.7) takes about

$$\frac{1}{2}\,n^2mk^2 + \frac{1}{6}\,n^3k^3 \qquad (6.23)$$

operations.

Thus we see that our method is much slower than that of Zellner [23] for m >> k. When m = 2k, Zellner's method is about twice faster than that of ours. But our method is numerically stable whereas Zellner's is not. Our method will be far more competitive if we could solve (6.13) efficiently.

CHAPTER 7

PARAMETER ESTIMATION IN A DYNAMICAL SYSTEM.

7.1 Introduction.

We have seen in Section 2.4.3 that in estimating the parameters of a dynamical system one often needs to obtain the least squares estimate of a linear model of the form

$$y = Cx + Bv\,;\; E(v) = 0\;,\; E(vv^T) = \sigma^2 I \qquad (7.1)$$

where

$$\underset{\sum_{i=1}^{t} m_i\times 1}{y} = \begin{bmatrix} y_1\\ y_2\\ \vdots\\ y_t\end{bmatrix},\; \underset{\sum_{i=1}^{t} m_i\times n}{C} = \begin{bmatrix} C_1\\ C_2\\ \vdots\\ C_t\end{bmatrix},\; \underset{\sum_{i=1}^{t} m_i\times\sum_{i=1}^{t} k_i}{B} = \begin{bmatrix} B_1 & & & \\ & B_2 & & \\ & & \ddots & \\ & & & B_t\end{bmatrix},\; \underset{\sum_{i=1}^{t} k_i\times 1}{v} = \begin{bmatrix} v_1\\ v_2\\ \vdots\\ v_t\end{bmatrix} \qquad (7.2)$$

y_i is a known m_i dimensional vector, C_i is a known matrix of dimension $m_i\times n$ and B_i is a full column rank matrix of rank k_i which is known.

The problem given in (7.1) can be reformulated as

$$\underset{v,x}{\text{minimize}}\; v^T v \quad \text{subject to} \quad y = Cx + Bv\,. \qquad (7.3)$$

We will solve (7.3) by using the fast step by step algorithm [18] for solving a general linear model discussed in Chapter 3. The advantage of using this algorithm lies in the fact that it reduces the system, step by step, affecting only a limited portion of the system at a time [18]. We will also give an operation count of the algorithm for a particular case.

../83

7.2 Method of solution

For simplicity we will assume that every B_i in the matrix B is an $m_i \times m_i$ nonsingular matrix. The algorithm that will be presented on the Section 7.3 will also work even if B is non square.

To begin with, let us first write

$$[\bar{y}_1, \bar{C}_1, \bar{B}_1] \equiv [y_1, C_1, B_1] \ . \tag{7.4}$$

By applying orthogonal rotations from left and right, it is possible to transform the initial data $[\bar{y}_1, \bar{C}_1, \bar{B}_1]$ to the following form:

$$Q_1^T [\underbrace{\bar{y}_1}_{1}, \underbrace{\bar{C}_1}_{n}, \underbrace{\bar{B}_1}_{m_1}] \begin{bmatrix} 1 & & \\ & I & \\ & & P_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & L_1^{(1)} & 0 \\ \underbrace{y_1^{(1)}}_{1} & \underbrace{C_1^{(1)}}_{n} & \underbrace{L_{21}^{(1)}}_{m_1-1} & \underbrace{B_1^{(1)}}_{n+1} \end{bmatrix} \begin{array}{l} \}m_1-n-1 \\ \}n+1 \end{array} \tag{7.5}$$

where Q_1 and P_1 are products of Givens orthogonal plane rotations and $L_1^{(1)}$ and $B_1^{(1)}$ are lower triangular matrices. Also all the elements above the main diagonal of $[y_1^{(1)}, C_1^{(1)}]$ are zero.

The reduction in (7.5) can be carried out in two stages. In the first stage $n(n+1)/2$ rotations are applied from the left to $[\bar{y}_1, \bar{C}_1]$ to zero out its upper diagonal part. We maintain the lower triangular form of $\bar{B}_1$ throughout the reduction by applying a rotation from the right to $\bar{B}_1$, whenever necessary.

In the second stage, we eliminate one diagonal of $[\bar{y}_1, \bar{C}_1]$ at each step keeping the form of $\bar{B}_1$ the same throughout, and this step is repeated until we arrive at the form given in (7.5). Therefore, the second stage has (m_1-n-1) steps and each step consists of $(n+1)$ pairs of rotation, a pair being one from the left to eliminate an element of the $[\bar{y}_1, \bar{C}_1]$ matrix, followed by one from the right to regain the triangular form of the $\bar{B}_1$ matrix.

Let

$$P_1^T = \begin{bmatrix} v_1^{(1)} \\ v_2 \\ \cdot \\ \cdot \\ \cdot \\ v_t \end{bmatrix} = \begin{bmatrix} v_{11}^{(1)} \\ \\ v^{(1)} \end{bmatrix}, \text{ say, } v_1^{(1)} = \begin{bmatrix} v_{11}^{(1)} \\ \cdot \\ v_{12}^{(1)} \end{bmatrix}, \ v^{(1)} = \begin{bmatrix} v_{12}^{(1)} \\ v_2 \\ \cdot \\ \cdot \\ \cdot \\ v_t \end{bmatrix}. \tag{7.6}$$

It is seen from the constraints in (7.3) and the transformations in (7.5) that

$$L_1^{(1)} v_{11}^{(1)} = 0 \tag{7.7}$$

which implies that

$$v_{11}^{(1)} = 0 \tag{7.8}$$

since $L_1^{(1)}$ is non-singular. Therefore, the problem (7.3) reduces to

$$\underset{v^{(1)},x}{\text{minimize}} \ v^{(1)T} v^{(1)} \text{ subject to } y^{(1)} = C^{(1)}x+B^{(1)}v^{(1)} \tag{7.9}$$

where

$$y^{(1)} = \begin{bmatrix} y_1^{(1)} \\ y_2 \\ \vdots \\ y_t \end{bmatrix}, \quad C^{(1)} = \begin{bmatrix} C_1^{(1)} \\ C_2 \\ \vdots \\ C_t \end{bmatrix}, \quad B^{(1)} = \begin{bmatrix} B_1^{(1)} & & & \\ & B_2 & & \\ & & \ddots & \\ & & & B_t \end{bmatrix}.$$

Let us assume that

$$[\underbrace{\bar{y}_2}_{1}, \underbrace{\bar{C}_2}_{n}, \underbrace{\bar{B}_2}_{m_2+n+1}] = \begin{array}{cl} \left[\begin{array}{cccc} y_1^{(1)} & C_1^{(1)} & B_1^{(1)} & 0 \\ y_2 & C_2 & 0 & B_2 \end{array}\right] & \begin{array}{l} \}\ n+1 \\ \}\ m_2 \end{array} \\ \begin{array}{cccc} 1 & n & n+1 & m_2 \end{array} & \end{array} \tag{7.10}$$

The matrix $[\bar{y}_2,\bar{C}_2]$ is already in lower trapezoidal form and $\bar{B}_2$ is also lower triangular.

By applying Givens rotations to the data $[\bar{y}_2,\bar{C}_2,\bar{B}_2]$ from the left and right, as before, we obtain the following transformation

$$Q_2^T[\bar{y}_2,\bar{C}_2,\bar{B}_2] \begin{bmatrix} 1 & & \\ & I & \\ & & P_2 \end{bmatrix} = \begin{array}{cl} \left[\begin{array}{cccc} 0 & 0 & L_1^{(2)} & 0 \\ y_2^{(2)} & C_2^{(2)} & L_{21}^{(2)} & B_2^{(2)} \end{array}\right] & \begin{array}{l} \}\ m_2 \\ \}\ n+1 \end{array} \\ \begin{array}{cccc} 1 & n & m_2 & n+1 \end{array} & \end{array} \tag{7.11}$$

where Q_2 and P_2 are products of Givens plane rotations, $L_1^{(2)}$, $B_2^{(2)}$ are lower triangular and $[y_2^{(2)},C_2^{(2)}]$ has all zero elements above its diagonal.

···/86

The transformation in (7.11) can be carried out by eliminating one diagonal of $[\bar{y}_2, \bar{C}_2]$ at each step by applying (n+1) rotationsfrom its left while keeping the form of B the same throughout. m_2 such steps will produce the form given in (7.11).

Let

$$P_2^T v^{(1)} = \begin{bmatrix} v_2^{(2)} \\ v_3 \\ \cdot \\ \cdot \\ \cdot \\ v_t \end{bmatrix} = \begin{bmatrix} v_{11}^{(2)} \\ \\ v^{(2)} \end{bmatrix} \text{, say } v_2^{(2)} = \begin{bmatrix} v_{11}^{(2)} \\ \\ v_{12}^{(2)} \end{bmatrix}, \quad v^{(2)} = \begin{bmatrix} v_{12}^{(2)} \\ v_3 \\ \cdot \\ \cdot \\ \cdot \\ v_t \end{bmatrix} \qquad (7.12)$$

Then from the constraints in (7.9) and the transformation given in (7.11) we find, as before,

$$L_1^{(2)} \; v_{11}^{(2)} = 0 \qquad (7.13)$$

which implies

$$v_{11}^{(2)} = 0 \; . \qquad (7.14)$$

Hence the system given in (7.0) is further reduced to the form

$$\underset{v^{(2)},x}{\text{minimize}} \; v^{(2)T} v^{(2)} \text{ subject to } y^{(2)} = C^{(2)}x + B^{(2)}v^{(2)} \qquad (7.15)$$

where

$$y^{(2)} = \begin{bmatrix} y_2^{(2)} \\ y_3 \\ \cdot \\ \cdot \\ \cdot \\ y_t \end{bmatrix}, \; C^{(2)} = \begin{bmatrix} C_2^{(2)} \\ C_3 \\ \cdot \\ \cdot \\ \cdot \\ C_t \end{bmatrix}, \text{ and } B^{(2)} = \begin{bmatrix} B_2^{(2)} & & & & \\ & B_3 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & B_t \end{bmatrix} .$$

We next work with the system

$$[\bar{y}_3,\bar{C}_3,\bar{B}_3] = \begin{array}{c} \left[\begin{array}{cccc} y_2^{(2)} & C_2^{(2)} & B_2^{(2)} & 0 \\ y_3 & C_3 & 0 & B_3 \end{array}\right] \begin{array}{l} \}\ n+1 \\ \}\ m_3 \end{array} \\ \begin{array}{cccc} \underbrace{}_{1} & \underbrace{}_{n} & \underbrace{}_{n+1} & \underbrace{}_{m_3} \end{array} \end{array} \qquad (7.16)$$

Continuing the process, at the end, the problem given in (7.3) reduces to the following form:

$$\underset{v^{(t)},x}{\text{minimize}}\ v^{(t)^T} v^{(t)} \text{ subject to } y_t^{(t)} = C_t^{(t)} x + B_t^{(t)} v^{(t)} \qquad (7.17)$$

where $[y_t^{(t)}, C_t^{(t)}]$ and $B_t^{(t)}$ are both lower triangular matrices of dimension (n+1) by (n+1). Thus the original problem is reduced to a small generalized least squares problem which can be solved very easily.

We find that the method is truly sequential in terms of blocks and we do not need to store any of the orthogonal matrices. Moreover, at any stage, one can estimate x_i by solving the generalized least squares problem

$$\underset{v^{(i)},x_i}{\text{minimize}}\ v^{(i)^T} v^{(i)} \text{ subject to } y_i^{(i)} = C_i^{(i)} x_i + B_i^{(i)} v^{(i)} \qquad (7.18)$$

and thus a check can be provided on whether x is settling down or not.

7.3 Algorithm of the method.

The algorithm of the method just described is as follows:

```
i) reduce $[y_1, C_1, B_1]$ to $\begin{bmatrix} 0 & 0 & L_1^{(1)} & 0 \\ y_1^{(1)} & C_1^{(1)} & L_{21}^{(1)} & B_1^{(1)} \end{bmatrix}$

ii) for i := 2,3,...,t do

    a) form $[\bar{y}_i, \bar{C}_i, \bar{B}_i] = \begin{bmatrix} y_{i-1}^{(i-1)} & C_{i-1}^{(i-1)} & B_{i-1}^{(i-1)} & 0 \\ y_i & C_i & 0 & B_i \end{bmatrix}$

    b) reduce $[\bar{y}_i, \bar{C}_i, \bar{B}_i]$ to $\begin{bmatrix} 0 & 0 & L_1^{(i)} & 0 \\ y_i^{(i)} & C_i^{(i)} & L_{21}^{(i)} & B_i^{(i)} \end{bmatrix}$

iii) reduce $[y_t^{(t)}, C_t^{(t)}, B_t^{(t)}]$ to $\begin{bmatrix} 0 & 0 & L_1 & 0 & 0 \\ \eta & 0 & g^T & \mu & 0 \\ z & R^T & L_{21} & r & L_2 \end{bmatrix}$

     , $R^T$ is a full row rank matrix.

iv) solve $\begin{bmatrix} \eta & 0 \\ z & R^T \end{bmatrix} \begin{bmatrix} \nu \\ \hat{x} \end{bmatrix} = \begin{bmatrix} \mu \\ r \end{bmatrix}$ for $\nu$ and $\hat{x}$.
```

7.4 Operation count.

We will assume for simplicity that each $C_i, i=1,\ldots,t$ in the matrix C is of the dimension $m \times n$ and each B_i is lower triangular. We will consider 4 multiplication Givens rotations only.

We find from the algorithm given in Section 7.3 that steps (i) and (ii) take most of the time for computations.

Step (i) takes about

$$2m^2n + 4m\,n^2 - \frac{8}{3}\,n^3 \qquad (7.19)$$

operations.

For each i in step (ii) the operation count is about

$$2m^2n + 6mn^2 \; . \qquad (7.20)$$

Therefore, the total operations for step (ii) is given by

$$2(t-1)m^2n + 6(t-1)mn^2 \qquad (7.21)$$

Since the other steps of the algorithm need fewer operations in comparison to (7.19) and (7.21), we can say that the total number of operations necessary to implement the algorithm is of the order of

$$2tm^2n + 6tmn^2 \; . \qquad (7.22)$$

From Section 2.4.3.2 we see that the number of operations required to solve the same problem using the algorithm given by Paige [16], is of the order of

$$\frac{1}{2}\,tm^2n + tmn^2 \; . \qquad (7.23)$$

From (7.22) and (7.23) we find that the method presented in this Chapter is slower than the one described in Section 2.4.3. But the present algorithm is very general and it does not fail when the variance-covariance matrix of the noise term is singular, e.g. when the number of observations is less than the number of elements in the state vector. The algorithm pre-

.../90

sented by Paige [16], discussed in the Section 2.4.3, is not numerically stable when the variance-covariance matrix is ill conditioned [12]. The algorithm of Grove et al [9] is also very poor numerically [16]. The algorithm presented here requires very little storage. The method is basically sequential and affects only a limited part of the system at a time. Therefore, it is capable of solving a very large system.

We can make the method more competitive by using stabilized nonunitary transformations in place of Givens plane rotations from the left of the system. This technique is much faster [18]. If we use stabilized nonunitary transformations from the left and fast 2 multiplication Givens rotations from the right then the total number of operations needed to implement the algorithm is of the order of

$$m^2nt + 2mn^2t \; . \qquad (7.24)$$

.../91

CHAPTER 8

COMMENTS AND CONCLUSIONS.

In this thesis we have considered various types of generalized linear least squares problems. Our main aim has been to develop reliable numerically stable algorithms which are computationally efficient in solving different types of generalized least squares problems.

In Chapter 3 we have presented a fast numerically stable algorithm to solve a generalized least squares problem which is based on the work carried out by Paige [18]. This algorithm is more efficient than the one presented by Paige [17] because the method reduces the size of the system as the reduction progresses. The speed of the computation can be made faster by considering stabilized nonunitary transformations from the left of the system and fast square root free Givens rotations from the right [18].

While solving a generalized least squares problem, it is often necessary to obtain the minimum 2-norm solution of a structured underdetermined system. In Chapter 4 we have presented an efficient and apparently numerically stable algorithm to solve such problems. The algorithm is particularly efficient if only a part of the solution is required. The algorithm is inexpensive and takes little storage to execute. Thus, it is capable of solving a large structured underdetermined system. The entire minimum 2-norm solution can also be found if required, but a large part of the original matrix must be retained in order to do that.

In Chapter 5 we have solved a generalized least squares problem which arises out of a controlled experiment. We have used the algorithm described in Chapter 4 to obtain the solution of the problem efficiently. The algorithm is very fast when the variance-covariance matrices of the noise term are the same for different experiments.

Chapter 6 describes a numerically stable method to estimate the parameters of a set of different ordinary linear models whose noise terms are correlated. Though the algorithm is general, it requires large storage. There may exist a better but less obvious approach that we have not found.

In Chapter 7 we have presented an algorithm, based on the work of Paige [18], which will solve a large generalized least squares problem arising out of the problem of parameter estimation in a dynamical system. The method utilizes the technique described in Chapter 3. The method is basically sequential and affects only a limited portion of the system at a time. The method is very fast and needs very little storage to operate. Therefore, the algorithm is capable of solving large systems very efficiently.

The algorithms developed in Chapters 5, 6 and 7 can further be made computationally faster by using stabilized nonunitary transformations and fast square root free Givens rotations [18]. We have compared our algorithms with the existing methods on the basis of 4 multiplication Givens rotations which are used to reduce a system.

APPENDIX A

PROCEDURE GLSQUARES

Procedure GLSQUARES solves a generalized least squares problem of the form

$$\min_{v,x} v^T v \text{ subject to } y = Cx + Bv$$

where y is a given m vector, C is an m×n matrix which is known and B is a known full column rank lower trapezoidal matrix of dimension m×k where $V = BB^T$. The procedure is based on the algorithm presented in Chapter 3.

The procedure has been written in ALGOLW and has been tested on an IBM/370 computer. The results have been checked against the IMSL subroutine LLSQAR which provides the solution of an overdetermined system of linear equations. LLSQAR solves a system of the form

$$\begin{bmatrix} W & C \\ C^T & 0 \end{bmatrix} \begin{bmatrix} r \\ \hat{x} \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix}$$

where $\hat{x}$ is the desired solution [12] , r is the residual and $W = BB^T$. Our results are based on double precision computations.

We have considered the following examples:

I. C with full column rank, B ill conditioned,

II. C with less than full column rank, B well conditioned,

III. C with less than full column rank, B non square with full column rank, and

IV. a wrong model in which the procedure responded with an appropriate message.

.../94

Example I

$m = 4$, $n = 3$, $k = 4$, C of full column rank, B ill conditioned [12].

$$C = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \\ 5 & 6 & 7 \\ 3 & 4 & 6 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & & & \\ 2 & 3 & & \\ 4 & 5 & 10^{-5} & \\ 7 & 8 & 9 & 10 \end{bmatrix}, \quad y = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

The estimate $\hat{x}$

GLSQUARES	LLSQAR
0.314099920054512	0.314099920054513
-0.334417273202255	-0.334417273202264
0.441690628763596	0.441690628763602

$\|v\| = 0.102785269022065$

$\|y-Cx-Bv\| = 4.04365493431965 \times 10^{-16}$

.../ 95

Example II

$m = 8$, $n = 5$, $k = 8$, C of less than full column rank, B well conditioned.

$$C = \begin{bmatrix} 22 & 10 & 2 & 3 & 7 \\ 14 & 7 & 10 & 0 & 8 \\ -1 & 13 & -1 & -11 & 3 \\ -3 & -2 & 13 & -2 & 4 \\ 9 & 8 & 1 & -2 & 4 \\ 9 & 1 & -7 & 5 & -1 \\ 2 & -6 & 6 & 5 & 1 \\ 4 & 5 & 0 & -2 & 2 \end{bmatrix}, \text{ B=I, an identity matrix, } y = \begin{bmatrix} -1 \\ 2 \\ 1 \\ 4 \\ 0 \\ -3 \\ 1 \\ 0 \end{bmatrix}$$

The estimate $\hat{x}$

GLSQUARES	LLSQAR
-0.0833333333333333	-0.0833333333333336
$-1.38777878078145 \times 10^{-17}$	$0.264086629167334 \times 10^{-16}$
0.25	0.250000000000001
-0.0833333333333332	-0.0833333333333336
0.0833333333333333	0.0833333333333336

$\|v\| = 8.32667268468867 \times 10^{-17}$,

$\|y-Cx-Bv\| = 2.91968996652571 \times 10^{-15}$

../ 96

Example III

$m = 6$, $n = 5$, $k = 3$, C of less than full column rank, B non-square with full column rank.

$$C = \begin{bmatrix} -74 & 80 & 18 & -11 & -4 \\ 14 & -69 & 21 & 28 & 0 \\ 66 & -72 & -5 & 7 & 1 \\ -12 & 66 & -30 & -23 & 3 \\ 3 & 8 & -7 & -4 & 1 \\ 4 & -12 & 4 & 4 & 0 \end{bmatrix}, B = \begin{bmatrix} 1 & & \\ 2 & 3 & \\ 4 & 5 & 6 \\ 7 & 9 & 3 \\ 1 & 4 & 5 \\ 5 & 3 & 6 \end{bmatrix}, y = \begin{bmatrix} 30 \\ -61 \\ -56 \\ 69 \\ 10 \\ -12 \end{bmatrix}$$

The estimate $\hat{x}$

GLSQUARES	LLSQAR
-8.45393615810305	-8.45393615810469
-7.97368463534777	-7.97368463534988
-3.56383765775711	-3.56383765775447
-14.92991315025910	-14.9299131502655
14.44875439787700	14.4487543978946

$\|v\| = 0.0189702679770842$

$\|y-Cx-Bv\| = 3.67163024571173 \times 10^{-13}$

.../97

Example IV

$m = 6$, $n = 2$, $k = 6$, C of full column rank, B singular.

$$C = \begin{bmatrix} 1 & 3 \\ 4 & 1 \\ 5 & 2 \\ 2 & 7 \\ 6 & 10 \\ 1 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & & & & & \\ 2 & 3 & & & & \\ 4 & 5 & 6 & & & \\ 0 & 0 & 0 & 0 & & \\ 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad y = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

The residual is 4.59787397898538

It is easy to verify that y does not lie in the space spanned by the columns of $[C,B]$.

.../98

```
BEGIN
 INTEGER M,N,K;

 PROCEDURE GLSQUARES(INTEGER VALUE M,N,K);
 BEGIN

  COMMENT

     SOLVING A GENERALIZED LEAST SQUARES PROBLEM.THE PROBLEM IS

                T
        MIN. V V  SUBJECT TO Y=CX+BV
        V,X

     WHERE C IS M BY N MATRIX WHICH MAY NOT BE OF FULL COLUMN RANK,
     B IS M X K MATRIX AND Y IS A KNOWN M-VECTOR,X IS AN UNKNOWN N-
     VECTOR AND V IS A K-VECTOR.IN PRACTICE B CAN BE NON-SQUARE.

     THE DESCRIPTIONS OF THE OTHER VARIABLES ARE GIVEN BELOW:

     V          : AN ARRAY CONTAINING THE VECTOR V.
     P          : THE ORTHOGONAL MATRIX MULTIPLIED TO THE MATRIX
                  C FROM THE RIGHT.
     TEMPC      : A COPY OF THE INPUT MATRIX C.ORIGINAL C IS USED TO
                  COMPUTE THE RESIDUE OF THE SYSTEM.
     TEMPB      : A COPY OF THE INPUT MATRIX B.
     TEMPY      : A COPY OF THE INPUT ARRAY Y.
     TOL        : AN ELEMENT OF A MATRIX IS NONZERO IF ITS
                  MAGNITUDE IS GREATER THAN 'TOL'.
     STARTCOLUMN: INDICATES THAT  COLUMN 1 TO STARTCOLUMN-1 OF
                  THE MATRIX B DO NOT HAVE ANY EFFECT ON THE SOLUTION
                  X.;


   LONG REAL ARRAY C(1::M,1::N);
   LONG REAL ARRAY B(1::M,1::K);
   LONG REAL ARRAY Y(1::M);
   LONG REAL ARRAY X(1::N);
   LONG REAL ARRAY V(1::K);
   LONG REAL ARRAY P(1::K,1::K);
   LONG REAL ARRAY TEMPC(1::M,1::N);
   LONG REAL ARRAY TEMPB(1::M,1::K);
   LONG REAL ARRAY TEMPY(1::M);
   INTEGER  ROWPLUSONE,COUNT,REDUCTION,ROW,COL;STARTCOLUMN;
   LONG REAL CC,SS,TEMP,MEW,TOL;
   LOGICAL CONTINUE;


    COMMENT

     THE PROCEDURE IS FOR APPLYING GIVENS PLANE ROTATIONS.LEFT AND
     RIGHT ROTATIONS ARE INDICATED BY THE PARAMETER 'SWITCH';
```

```
PROCEDURE GIVENS(LONG REAL VALUE Z1,Z2;INTEGER VALUE SWITCH);
COMMENT

  IF SWITCH=0 THEN THE GIVENS ROTATION IS THE LEFT ONE
  AND IF SWITCH=1 THEN THE ROTATION IS APPLIED FROM RIGHT.
  IN THE FIRST CASE Z1 IS ELIMINATED WHEREAS IN THE SECOND
  CASE Z2 IS ELIMINATED;

BEGIN
  LONG REAL GAMA;
  GAMA:=Z1*Z1+Z2*Z2;
  GAMA:=LONGSQRT(GAMA);
  IF SWITCH=0 THEN
  BEGIN
    CC:=Z2/GAMA;
    SS:=Z1/GAMA
  END
  ELSE
  BEGIN
    CC:=Z1/GAMA;
    SS:=Z2/GAMA
  END
END GIVENS;


COMMENT

  THE ORTHOGONAL PLANE ROTATION IS APPLIED FROM THE LEFT TO C.
  THE VARIABLES 'CC' & 'SS' ARE EVALUATED IN THE PROCEDURE
  GIVENS;

PROCEDURE UPDATEC(INTEGER VALUE ROW1,ROW2,COL;LONG REAL
                                                ARRAY C(*,*));
BEGIN
LONG REAL TEMP;
  FOR I:=1 UNTIL COL DO
  BEGIN
    TEMP:=-CC*C(ROW1,I)+SS*C(ROW2,I);
    C(ROW2,I):=SS*C(ROW1,I)+CC*C(ROW2,I);
    C(ROW1,I):=TEMP
  END
END UPDATEC;


COMMENT

  UPDATING Y BY MULTIPLYING THE ORTHOGONAL MATRIX FROM
  LEFT TO THE VECTOR Y;

PROCEDURE UPDATEY(INTEGER VALUE ROW1,ROW2;LONG REAL ARRAY Y(*));

BEGIN
LONG REAL TEMP;
  TEMP:=-CC*Y(ROW1)+SS*Y(ROW2);
  Y(ROW2):=SS*Y(ROW1)+CC*Y(ROW2);
```

.../100

```
   Y(ROW1):=TEMP
 END UPDATEY;


 COMMENT

   THE ORTHOGONAL  ROTATION MATRIX IS MULTIPLIED TO  THE MATRIX B.
   FROM THE LEFT;

 PROCEDURE LEFTUPDATEB(INTEGER VALUE ROW1,ROW2,RANKB,STARTCOLUMN;
                                         LONG REAL ARRAY B(*,*));

 BEGIN
 LONG REAL  TEMP;
   IF STARTCOLUMN := RANKB THEN
   BEGIN
     FOR I:=STARTCOLUMN UNTIL (IF ROW1 RANKB THEN ROW2 ELSE RANKB)
                                                                DO
     BEGIN
       TEMP:= -CC*B(ROW1,I)+SS*B(ROW2,I);
       B(ROW2,I):=SS*B(ROW1,I)+CC*B(ROW2,I);
       B(ROW1,I):=TEMP
     END
   END
 END LEFTUPDATEB;


 COMMENT

   THE ORTHOGONAL ROTATIONS ARE APPLIED FROM RIGHT TO THE
   MATRIX B TO MAINTAIN THE FORM OF B WHICH IS LOWER TRAPEZOIDAL
   WHEN BROWS  IS NOT EQUAL  TO RANKB OTHERWISE IT IS LOWER
   TRIANGULAR;

 PROCEDURE RIGHTUPDATEB(INTEGER VALUE COL1,COL2,BROWS;
                                              LONG REAL ARRAY B(*,*));

 BEGIN
 LONG REAL  TEMP;
 INTEGER ROW1;
   ROW1:=COL1;
   FOR I:=ROW1 UNTIL BROWS DO
   BEGIN
     TEMP:=CC*B(I,COL1)+SS*B(I,COL2);
     B(I,COL2):=SS*B(I,COL1) CC*B(I,COL2);
     B(I,COL1):=TEMP
   END
 END RIGHTUPDATEB;

 COMMENT

 RESTORING THE FORM OF THE MATRIX B WHICH IS GENERALLY LOWER
 TRAPEZOIDAL .(COL1,COL2)ELEMENT OF B IS MADE ZERO AND THE
 WEIGHT IS GIVEN TO (COL1,COL1)ELEMENT OF B;
```

.../101

```
PROCEDURE RESTORER(INTEGER VALUE COL1,COL2,BROWS,BCOLS;
                   LONG REAL ARRAY B(*,*);LONG REAL ARRAY P(*,*));

BEGIN
  LONG REAL TEMP;
  IF COL2 <= BCOLS THEN
  IF ABS(B(COL1,COL2)) > TOL THEN
  BEGIN
    GIVENS(B(COL1,COL1),B(COL1,COL2),1);
    RIGHTUPDATER(COL1,COL2,BROWS,B);
    FOR I:=1 UNTIL BCOLS DO
    BEGIN
      TEMP:=CC*P(COL1,I)+SS*P(COL2,I);
      P(COL2,I):=SS*P(COL1,I)-CC*P(COL2,I);
      P(COL1,I):=TEMP
    END
  END
END RESTORER;


COMMENT .

  THIS PROCEDURE  WILL LEAVE BLANKLINE WHILE PRINTING THE
  NUMBER OF LINES IS INDICATED BY L;

PROCEDURE BLANKLINE(INTEGER VALUE L);
BEGIN
  FOR I:=1 UNTIL L DO WRITE(" ")
END BLANKLINE;


COMMENT

READING THE INPUT DATA AND PRINTING THEM OUT.IF THE SYSTEM
IS UNDERDETERMINED IT STOPS AFTER PRINTING THE APPROPRIATE
MESSAGE;

IF M < N THEN
BEGIN
  WRITE("***THE SYSYTEM IS UNDERDETERMINED***");
  GO TO STOP
END;
FOR I:=1 UNTIL M DO
FOR J:=1 UNTIL N DO READON(C(I,J));
FOR I:=1 UNTIL K-1 DO
FOR J:=I+1 UNTIL K DO B(I,J):=0;
FOR I:=1 UNTIL M DO
FOR J:=1 UNTIL (IF I > K THEN K ELSE I) DO READON(B(I,J));
FOR I:=1 UNTIL M DO READON(Y(I));
WRITE("THE NUMBER OF OBSERVATION  ",M);
WRITE(" ");
WRITE(" THE NUMBER OF PARAMETERS TO BE ESTIMATED ",N);
WRITE(" ");
WRITE("THE DIMENSION OF THE MATRIX B IS  ",M," X ",K);
BLANKLINE(4);
WRITE("THE MATRIX C");
```

.../102

```
BLANKLINE(2);
FOR I:=1 UNTIL M DO
BEGIN
  BLANKLINE(2);
  FOR J:=1 UNTIL N DO WRITEON(SHORT(C(I,J)))
END;
BLANKLINE(2);
WRITE("THE VECTOR Y");
BLANKLINE(2);
FOR I:=1 UNTIL M DO WRITE(Y(I));
BLANKLINE(5);
WRITE("THE MATRIX B");
BLANKLINE(2);
FOR I:=1 UNTIL M DO
BEGIN
  BLANKLINE(1);
  FOR J:=1 UNTIL K DO WRITEON(SHORT(B(I,J)))
END;

COMMENT

  INITIALIZING P TO AN IDENTITY MATRIX;

FOR I:=1 UNTIL K DO
BEGIN
  FOR J:=I UNTIL K DO P(I,J):=P(J,I):=0;
  P(I,I):=1
END;

COMMENT

  KEEPING A COPY OF THE INPUT DATA;

FOR I:=1 UNTIL M DO
BEGIN
  FOR J:=1 UNTIL N DO TEMPC(I,J):=C(I,J);
  FOR J:=1 UNTIL K DO TEMPB(I,J):=B(I,J);
  TEMPY(I):=Y(I)
END;

COMMENT

    REDUCING (Y,C) TO LOWER TRAPEZOIDAL FORM AND AT THE SAME
    TIME KEEPING THE FORM OF B THE SAME THROUGHOUT;

TOL:=1'-14;
STARTCOLUMN:=1;
FOR COL:=N STEP -1 UNTIL (IF M > N THEN 1 ELSE 2) DO
FOR ROW:=1 UNTIL (IF M > N THEN COL ELSE COL-1) DO
BEGIN
  ROWPLUSONE:=ROW+1;
  IF ABS(C(ROW,COL)) > TOL THEN
  BEGIN
    GIVENS(C(ROW,COL),C(ROWPLUSONE,COL),0);
    UPDATEY(ROW,ROWPLUSONE,Y);
    UPDATEC(ROW,ROWPLUSONE,COL,C);
    LEFTUPDATER(ROW,ROWPLUSONE,K,STARTCOLUMN,B);
    RESTOREB(ROW,ROWPLUSONE,M,K,B,P)
```

.../103

```
    END
  END;

  COMMENT

    THE MATRIX (Y,C) WILL NOW BE REDUCED TO THE FORM(O,E,L)WHERE
    (E,L) IS A LOWER TRIANGULAR MATRIX.WE WILL ALSO KEEP THE FORM OF
    B THE SAME THROUGHOUT;

  IF M N+1 THEN
  BEGIN
    FOR NO:=1 UNTIL M-N-1 DO
    BEGIN
      FOR COL:=N STEP -1 UNTIL 1 DO
      BEGIN
        ROW:=COL+NO;
        ROWPLUSONE:=ROW+1;
        IF ABS(C(ROW,COL)) > TOL THEN
        BEGIN
          GIVENS(C(ROW,COL),C(ROWPLUSONE,COL),O);
          UPDATEY(ROW,ROWPLUSONE,Y);
          UPDATEC(ROW,ROWPLUSONE,COL,C);
          LEFTUPDATEB(ROW,ROWPLUSONE,K,STARTCOLUMN,B);
          RESTOREB(ROW,ROWPLUSONE,M,K,B,P)
        END
      END;
      IF ABS(Y(NO)) > TOL THEN
      BEGIN
        ROWPLUSONE:=NO+1;
        GIVENS(Y(NO),Y(NO+1),O);
        UPDATEY(NO,ROWPLUSONE,Y);
        LEFTUPDATEB(NO,ROWPLUSONE,K,STARTCOLUMN,B);
        RESTOREB(NO,ROWPLUSONE,M,K,B,P)
      END;
      STARTCOLUMN:=STARTCOLUMN+1
    END
  END;

  COMMENT

      (Y,C) HAS BEEN REDUCED TO (O,E,L) FORM.'E' IS A VECTOR AND
      L IS A LOWER TRIANGULAR MATRIX .NOW L WILL BE REDUCED TO
      FULL ROW RANK MATRIX BY APPLYING ROTATIONS FROM LEFT.
      THE RANK OF THE REDUCED MATRIX WILL BE GIVEN BY N-REDUCTION;

  REDUCTION:=0;
  ROW:=M;
  COL:=N;
  CONTINUE:=TRUE;
  WHILE(CONTINUE) DO
  BEGIN
    CONTINUE:=FALSE;

    WHILE((COL>O) AND(ABS(C(ROW,COL)) > TOL))DO
```

```
    BEGIN
      ROW:=ROW-1;
      COL:=COL-1
    END;
    IF COL>0 THEN REDUCTION:=REDUCTION+1;
    IF COL>1 THEN
    BEGIN
      CONTINUE:=TRUE;
      COUNT:=1;
      FOR I:=COL-1 STEP -1 UNTIL 1 DO
      BEGIN
        ROWPLUSONE:=ROW-COUNT+1;
        IF ABS(C(ROW-COUNT,I)) > TOL THEN
        BEGIN
          GIVENS(C(ROW-COUNT,I),C(ROWPLUSONE,I),0);
          UPDATEY(ROW-COUNT,ROWPLUSONE,Y);
          UPDATEC(ROW-COUNT,ROWPLUSONE,I,C);
          LEFTUPDATEB(ROW-COUNT,ROWPLUSONE,K,STARTCOLUMN,B);
          RESTOREB(ROW-COUNT,ROWPLUSONE,M,K,B,P);
        END;
        COUNT:=COUNT+1
      END
    END;
    IF COL>0 THEN
    BEGIN
      ROWPLUSONE:=M-N+REDUCTION;
      IF ABS(Y(ROWPLUSONE-1)) > TOL THEN
      BEGIN
        GIVENS(Y(ROWPLUSONE-1),Y(ROWPLUSONE),0);
        UPDATEY(ROWPLUSONE-1,ROWPLUSONE,Y);
        LEFTUPDATEB(ROWPLUSONE-1,ROWPLUSONE,K,STARTCOLUMN,B);
        RESTOREB(ROWPLUSONE-1,ROWPLUSONE,M,K,B,P);
        STARTCOLUMN:=STARTCOLUMN+1
      END
    END;
    COL:=COL-1
  END;

  COMMENT

    TESTING THE CONSISTENCY OF THE MODEL WHEN REDUCTION>0;

  IF M > N THEN
  BEGIN
    ROW:=M-N+REDUCTION;
    IF ROW <= K THEN
    BEGIN
      IF ABS(B(ROW,ROW)) > TOL THEN
      BEGIN
        IF ABS(Y(ROW)) > TOL THEN
        BEGIN
          WRITE("***THE SYSTEM IS INCONSISTENT***");
          BLANKLINE(2);
          WRITE("   THE RESIDUAL IS  ",Y(ROW));
          GO TO STOP
        END
```

```
      ELSE MEW:=0
    END
    ELSE MEW:=Y(ROW)/B(ROW,ROW);
    FOR I:=ROW+1  UNTIL M DO
    Y(I):=Y(I)-MEW*B(I,ROW);
    FOR I:=1 UNTIL N DO V(I):=B(ROW,I)*MEW
  END
  ELSE IF ABS(Y(ROW)) > TOL THEN
    BEGIN
      WRITE("*** THE SYSTEM IS INCONSISTENT***");
      BLANKLINE(3);
      WRITE("    THE RESIDUAL IS  ",Y(ROW));
      GO TO STOP
    END
 END;

COMMENT

    AT THIS STAGE C IS A LOWER TRAPEZOIDAL MATRIX OF FULL ROW RANK
    IF REDUCTION > 0. C IS NOW REDUCED TO LOWER TRIANGULAR FORM
    FROM LOWER TRAPEZOIDAL FORM BY APPLYING GIVENS ROTATIONS FROM
    RIGHT.THESE ROTATIONS ARE MULTIPLIED TO OBTAIN THE ORTHOGONAL
    MATRIX Q WHICH WHEN MULTIPLIED TO LOWER TRAPEZOIDAL MATRIX C
    FROM RIGHT WILL YIELD THE LOWER TRIANGULAR MATRIX;

BEGIN
  LONG REAL ARRAY Q(1::N,1::N);
  INTEGER STEPS;
  FOR I:=1 UNTIL N DO
  FOR J:=1 UNTIL N DO IF I=J THEN Q(I,J):=1 ELSE Q(I,J):=0;
  IF REDUCTION   0 THEN
  BEGIN
    STEPS:=0;
    FOR ROW:=M-N+REDUCTION+1 UNTIL M DO
    BEGIN
      FOR J:=REDUCTION STEP -1 UNTIL 1 DO
      BEGIN
        COL:=STEPS+J+1;
        IF ABS(C(ROW,COL)) > TOL THEN
        BEGIN
          GIVENS(C(ROW,COL-1),C(ROW,COL),1);
          FOR I:=ROW UNTIL M DO
          BEGIN
            TEMP:=CC*C(I,COL-1)+SS*C(I,COL);
            C(I,COL):=SS*C(I,COL-1)-CC*C(I,COL);
            C(I,COL-1):=TEMP
          END;
          FOR I:=1 UNTIL N DO
          BEGIN
            TEMP:=CC*Q(I,COL-1)+SS*Q(I,COL);
            Q(I,COL):=SS*Q(I,COL-1)-CC*Q(I,COL);
            Q(I,COL-1):=TEMP
          END
        END
      END;
      STEPS:=STEPS+1
    END
  END;
  BLANKLINE(4);
```

.../106

```
WRITE("THE RANK OF THE MATRIX C IS  ",N-REDUCTION);
BLANKLINE(2);
WRITE("THE TRANSFORMED C");
FOR I:=1 UNTIL M DO
BEGIN
  BLANKLINE(2);
  FOR J:=1 UNTIL N DO WRITEON(C(I,J))
END;
BLANKLINE(2);
WRITE("THE TRANSFORMED VECTOR Y");
BLANKLINE(2);
FOR I:=1 UNTIL M DO  WRITE("  ",Y(I));
BLANKLINE(10);
WRITE("THE TRANSFORMED B");
FOR I:=1 UNTIL M DO
BEGIN
  BLANKLINE(2);
  FOR J:=1 UNTIL K DO WRITEON(SHORT(B(I,J)))
END;

COMMENT

   EVALUATING THE VECTOR X;

X(1):=Y(M-N+REDUCTION+1)/C(M-N+REDUCTION+1,1);
FOR I:=2 UNTIL N-REDUCTION DO
BEGIN
  TEMP:=0;
  FOR J:=1 UNTIL I-1 DO
  TEMP:=TEMP+C(M-N+REDUCTION+I,J)*X(J);
  X(I):=(Y(M-N+REDUCTION+I)-TEMP)/C(M-N+REDUCTION+I,I)
END;

COMMENT

   OUR X:=Q*X;

IF REDUCTION ¬= 0 THEN
BEGIN
  FOR I:=1 UNTIL N-REDUCTION DO
  FOR J:=1 UNTIL N DO
  Q(J,I):=Q(J,I)*X(I);
  FOR I:=1 UNTIL N DO
  BEGIN
    TEMP:=0;
    FOR J:=1 UNTIL N-REDUCTION DO
    TEMP:=TEMP+Q(I,J);
    X(I):=TEMP
  END
END;
BLANKLINE(2);
WRITE("THE SOLUTIONS ARE");
FOR I:=1 UNTIL N DO WRITE(" ",X(I));
```

.../108

```
      COMMENT
        COMPUTING THE RESIDUE OF THE SYSTEM AND ALSO NORM OF THE VECTOR
        V;
      BEGIN
        LONG REAL TEMP1,TEMP2,TEMP3;
        TEMP1:=0;
      COMMENT
        COMPUTING ||Y-CX-BV||;
        FOR I:=1 UNTIL M DO
        BEGIN
          TEMP2:=0;
          FOR J:=1 UNTIL N DO TEMP2:=TEMP2+TEMPC(I,J)*X(J);
          TEMP3:=0;
          FOR J:=1 UNTIL K DO TEMP3:=TEMP3+TEMPB(I,J)*V(J);
          TEMP1:=TEMP1+(TEMPY(I)-(TEMP2+TEMP3))**2
        END;
        BLANKLINE(3);
        TEMP1:=LONGSQRT(TEMP1);
        WRITE("      NORM OF THE RESIDUE OF THE SYSTEM IS   ",TEMP1);
        COMMENT
          COMPUTING ||V||;
        TEMP2:=0;
        FOR J:=1 UNTIL K DO TEMP2:=TEMP2+V(J)**2;
        TEMP2:=LONGSQRT(TEMP2);
        BLANKLINE(3);
        WRITE("      NORM OF THE VECTOR V IS   ",TEMP2)
       END
     END;
  STOP:BLANKLINE(40)
  END GLSQUARES;
  COMMENT
    MAIN PROGRAM STARTS HERE.THE DIMENSIONS OF C AND B HAVE BEEN
    PASSED TO THE PROCEDURE GLSQUARES AS THE PARAMETERS;
  READ(M,N,K);
  GLSQUARES(M,N,K)
END.
```

.../ 108

APPENDIX B.

PROCEDURE MINNORM.

Procedure MINNORM obtains a part of the minimum 2-norm solutions of a structured underdetermined system based on the algorithm presented in Chapter 4. Let

$$y = F \begin{bmatrix} z_1 \\ \\ z_2 \end{bmatrix}$$

be the system. We are interested in obtaining $\hat{z}_1$ only.

The procedure has been written in ALGOLW and has been tested on an IBM/370 computer. The results have been checked against the solution obtained by solving the system without considering the structure. The results are based on double precision computations.

We have considered the following examples:

I. F with full row rank, and

II. F with less than full row rank.

.../109

EXAMPLE I.

F has full row rank.

$$F = \begin{bmatrix} G_1 & L_1 & 0 & 0 \\ G_2 & 0 & L_2 & 0 \\ G_3 & 0 & 0 & L_3 \end{bmatrix} = \left[\begin{array}{ccc|ccc|cccc|ccc}
10 & 1 & 2 & 3 & & & & & & & & & \\
1 & 9 & -1 & 2 & -3 & & & & & & & & \\
2 & -1 & 7 & 3 & -5 & 1 & & & & & & & \\
3 & 2 & 3 & 12 & -1 & 2 & & & & & & & \\
\hline
1 & 2 & 3 & & & & 1 & & & & & & \\
4 & 1 & 2 & & & & 2 & 3 & & & & & \\
5 & 6 & 7 & & & & 4 & 5 & 6 & & & & \\
3 & 4 & 6 & & & & 7 & 8 & 9 & 10 & & & \\
\hline
1 & 3 & 5 & & & & & & & & 1 & & \\
7 & 1 & 2 & & & & & & & & 4 & 5 & \\
2 & 1 & 3 & & & & & & & & 5 & 3 & 2
\end{array}\right], \quad y = \left[\begin{array}{c} 2 \\ 1 \\ 3 \\ 2 \\ \hline 6 \\ 7 \\ 8 \\ 9 \\ \hline 1 \\ 2 \\ 3 \end{array}\right]$$

The estimate $\hat{z}_1$

0.125710961300614

0.0747320776983494

0.283670434928731

.../ 110

EXAMPLE II

F has less than full row rank.

$$F=\begin{bmatrix} G_1 & L_1 & 0 \\ G_2 & 0 & L_2 \end{bmatrix}=\left[\begin{array}{ccc|ccc|cccc} 3 & 0 & -11 & -2 & & & & & & \\ 7 & 8 & 3 & 2 & 2 & & & & & \\ 2 & 10 & -1 & 0 & 1 & 12 & & & & \\ 22 & 14 & -1 & 4 & 5 & -12 & & & & \\ 10 & 7 & 13 & 5 & 3 & -10 & & & & \\ \hline 1 & 2 & 3 & & & & 1 & & & \\ 4 & 1 & 2 & & & & 2 & 3 & & \\ 5 & 6 & 7 & & & & 4 & 5 & 6 & \\ 3 & 4 & 6 & & & & 7 & 8 & 9 & 10 \end{array}\right], \quad y=\left[\begin{array}{c} 2 \\ 1 \\ 3 \\ 2 \\ -1.75 \\ \hline 6 \\ 7 \\ 8 \\ 9 \end{array}\right]$$

The estimate $\hat{z}_1$

0.416404256666899

0.611175083020191

0.419992141744106

..../111

```
BEGIN
  INTEGER ROWBLOCKS;

  PROCEDURE MINNORM(INTEGER VALUE ROWBLOCKS);
  BEGIN

    COMMENT.

      MINIMUM 2-NORM SOLUTION OF A STRUCTURED UNDERDETERMINED SYSTEM OF
      THE FORM

                          Y=(G,L)[Z  ]
                                 [ 1 ]
                                 [Z  ]
                                 [ 2 ]
      WHERE
                 G=[G ],  L=[L                ]
                   [ 1]     [ 1               ]
                   [G ]     [   L             ]
                   [ 2]     [    2            ]
                   [. ]     [      .          ]
                   [. ]     [        .        ]
                   [G ]     [     .     L     ]
                   [ N]     [            N    ]

      THE PROGRAM WILL OBTAIN THE SOLUTION OF THE VECTOR Z  ONLY.
                                                          I
      THE DESCRIPTIONS OF THE VARIABLES USED ARE GIVEN BELOW:

      ROWBLOCKS : NUMBER OF BLOCKS OF ROWS IN THE SYSTEM.
      M         : AN ARRAY CONTAINING THE NUMBER OF ROWS IN EACH BLOCK.
      N         : NUMBER OF COLUMNS IN THE MATRIX G.
      K         : AN ARRAY CONTAINING THE NUMBER OF COLUMNS IN EACH
                  MATRIX L.
                          I
      TOL       : AN ELEMENT OF A MATRIX IS NONZERO IF ITS MAGNITUDE
                  IS GREATER THAN 'TOL'.
      G         : THE MATRIX G  .
                               I
      L         : THE MATRIX L  .
                               I
      S         : IT IS A MATRIX OF DIMENSION N BY (N+M ).
                                                      I
                  IN THE BEGINNING IT IS (I,0).AFTER EACH ROW BLOCK
                  REDUCTION IT CONTAINS (Z    ,0) AT THE START OF NEXT
                  REDUCTION.               I+1
      Y         : THE VECTOR Y.
      D         : IT IS A VECTOR.D  =S W +S W +. . . +S W .
                                  I   1 1  2 2        I J
      W         : A VECTOR WHICH CONTAINS THE SOLUTION OF THE
                  TRANSFORMED SYSTEM.;
```

.../ 112

```
    INTEGER ARRAY M(1::ROWBLOCKS);
    INTEGER ARRAY K(1::ROWBLOCKS);
    INTEGER N,NEWN,MAX,SROWNO,ROWNO;
    LONG REAL TOL,CC,SS;

  COMMENT

    PRINTING L NUMBER OF BLANKLINES;

    PROCEDURE BLANKLINE(INTEGER VALUE L);
    BEGIN
      FOR J:=1 UNTIL L DO WRITE(" ")
    END BLANKLINE;

    COMMENT

     DATA ARE BEING READ;

    READ(N);
    WRITE("NUMBER OF COLUMNS IN THE MATRIX G IS",N);
    NEWN:=N;
    MAX:=0;
    TOL:=1'-14;
    FOR BLOCKS:=1 UNTIL ROWBLOCKS DO
    BEGIN
      READ (M(BLOCKS),K(BLOCKS));
      WRITE("BLOCK NO.",BLOCKS);
      WRITE("NO OF ROWS :M=",M(BLOCKS));
      WRITE("NO OF COLS OF G :N=",N);
      WRITE("NO OF COLUMNS OF L :K",K(BLOCKS));
      BLANKLINE(3);

      COMMENT

        THE MAXIMUM ROW DIMENSION OF THE BLOCKS IS OBTAINED IN ORDER
        TO DECLARE THE DIMENSION OF THE MATRIX S PROPERLY;

      IF MAX < M(BLOCKS) THEN MAX:=M(BLOCKS);
      IF N+K(BLOCKS)<M(BLOCKS) THEN
      BEGIN
        WRITE("THE BLOCK NO ",BLOCKS,"  IS OVERDETERMINED");
        GO TO STOP
      END
    END;
    BEGIN
      LONG REAL ARRAY S(1::N,1::NEWN+MAX);
      LONG REAL ARRAY D(1::N);
      FOR I:=1 UNTIL N DO
      BEGIN
       FOR J:=1 UNTIL NEWN+MAX DO
                S(I,J):=0;
       S(1,I):=1
      END;
      FOR I:=1 UNTIL N DO D(1):=0;
```

```
FOR BLOCKS:=1 UNTIL ROWBLOCKS DO
BEGIN
  LONG REAL  ARRAY G(1::M(BLOCKS),1::N);
  LONG REAL  ARRAY L(1::M(BLOCKS),1::K(BLOCKS));
  LONG REAL  ARRAY Y(1::M(BLOCKS));
  LONG REAL  ARRAY C(1::M(BLOCKS),1::NEWN+K(BLOCKS));
  LONG REAL  ARRAY W(1::MAX);
  INTEGER TOTALROW,TOTALCOL,ROW,COL,COLMINUS1,COUNT,REDUCTION;
  LOGICAL  CONTINUE;

COMMENT

    PROCEDURE FOR APPLYING GIVENS ROTATION.LEFT AND RIGHT ROTATIONS
    ARE INDICATED BY THE PARAMETER SWITCH. IF SWITCH=0 THEN IT IS
    THE LEFT ONE AND IF SWITCH=1 THEN IT IS THE RIGHT ONE.IN THE
    FIRST CASE Z1 IS ELIMINATED WHEREAS IN  SECOND CASE Z2 IS
    ELIMINATED;

 PROCEDURE GIVENS(LONG REAL VALUE Z1,Z2;INTEGER VALUE SWITCH);
 BEGIN
   LONG REAL GAMA;
   GAMA:=Z1*Z1+Z2*Z2;
   GAMA:=LONGSQRT(GAMA);
   IF SWITCH = 0 THEN
   BEGIN
     CC:=Z2/GAMA;
     SS:=-Z1/GAMA
   END
   ELSE
   BEGIN
     CC:=Z1/GAMA;
     SS:=Z2/GAMA
   END
 END GIVENS;


 COMMENT
   ORTHOGONAL ROTATIONS ARE APPLIED FROM THE RIGHT.THE MATRIX
   'MATRIX' IS UPDATED.COL1 AND COL2 AND ROWS FROM FIRST TO LAST
   ARE AFFECTED.;

 PROCEDURE UPDATERIGHT(INTEGER VALUE COL1,COL2,FIRST,LAST;
                              LONG REAL ARRAY MATRIX(*,*));
 BEGIN
   LONG REAL TEMP;
   FOR I:=FIRST UNTIL LAST DO
   BEGIN
     TEMP:=CC*MATRIX(I,COL1)+SS*MATRIX(I,COL2);
     MATRIX(I,COL2):=SS*MATRIX(I,COL1)-CC*MATRIX(I,COL2);
     MATRIX(I,COL1):=TEMP
   END
 END UPDATERIGHT;

 COMMENT

   ORTHOGONAL ROTATIONS ARE APPLIED FROM LEFT.ROW1 AND ROW2 OF
   THE MATRIX 'MATRIX' FROM COLUMN 1 TO COLUMN LAST IS AFFECTED.;
```

.../114

```
PROCEDURE UPDATELEFT(INTEGER VALUE ROW1,ROW2,LAST;LONG REAL
                                ARRAY MATRIX(*,*));
BEGIN
  LONG REAL TEMP;
  FOR I:=1 UNTIL LAST DO
  BEGIN
    TEMP:=-CC*MATRIX(ROW1,I)+SS*MATRIX(ROW2,I);
    MATRIX(ROW2,I):=SS*MATRIX(ROW1,I)+CC*MATRIX(ROW2,I);
    MATRIX(ROW1,I):=TEMP
  END
END UPDATELEFT;

COMMENT

  THE VECTOR Y IS UPDATED FOR LEFT ROTATION WHICH AFFECTS ROW1
  AND ROW2 ONLY.;


PROCEDURE UPDATEY(INTEGER VALUE ROW1,ROW2;LONG REAL ARRAY Y(*));
BEGIN
  LONG REAL TEMP;
  TEMP:=-CC*Y(ROW1)+SS*Y(ROW2);
  Y(ROW2):=SS*Y(ROW1)+CC*Y(ROW2);
  Y(ROW1):=TEMP
END UPDATEY;

COMMENT

  THE MATRIX A OF DIMENSION ROW BY COLUMN IS PRINTED OUT.;


PROCEDURE PRINT(LONG REAL ARRAY A(*,*);INTEGER VALUE ROW,COL);
BEGIN
  FOR I:=1 UNTIL ROW DO
  BEGIN
    WRITE(" ");
    FOR J:=1 UNTIL COL DO WRITEON(SHORT(A(I,J)))
  END;
  BLANKLINE(2)
END PRINT;

COMMENT

   INPUT DATA ARE BEING READ;

   FOR I:=1 UNTIL M(BLOCKS) DO
   FOR J:=1 UNTIL N DO READON(G(I,J));
   WRITE(" MATRIX G FOR BLOCK NUMBER",BLOCKS);
   BLANKLINE(3);
   PRINT(G,M(BLOCKS),N);
   FOR I:=1 UNTIL M(BLOCKS) DO
   FOR J:=1 UNTIL K(BLOCKS) DO READON(L(I,J));
```

.../115

```
BLANKLINE(2);
WRITE(" MATRIX L FOR BLOCK NUMBER",BLOCKS);
BLANKLINE(3);
PRINT(L,M(BLOCKS),K(BLOCKS));
FOR I:=1 UNTIL M(BLOCKS) DO READON(Y(I));
BLANKLINE(3);
WRITE("VECTOR Y FOR BLOCK",BLOCKS);
BLANKLINE(3);
FOR I:=1 UNTIL M(BLOCKS)DO WRITEON(Y(I));

COMMENT

  IF L IS NOT INPUTTED IN LOWER TRAPEZOIDAL FORM THEN IT IS
  MADE LOWER TRAPEZOIDAL FORM BY APPLYING ROTATIONS FROM
  THE RIGHT;

FOR ROW:=1 UNTIL K(BLOCKS)-1 DO
FOR COL:=K(BLOCKS) STEP -1 UNTIL ROW+1 DO
BEGIN
  IF ABS(L(ROW,COL)) > TOL THEN
  BEGIN
    COLMINUS1:=COL-1;
    GIVENS(L(ROW,COLMINUS1),L(ROW,COL),1);
    UPDATERIGHT(COLMINUS1,COL,ROW,M(BLOCKS),L)
  END
END;

COMMENT

  FORMING GZ.IN THE FIRST ROW BLOCK Z=I,I IS A UNIT MATRIX.;

IF BLOCKS> 1 THEN
BEGIN
  LONG REAL ARRAY TEMP(1::NEWN);
  LONG REAL SUM;
  FOR I:=1 UNTIL M(BLOCKS) DO
  BEGIN
    FOR J:=1 UNTIL NEWN DO
    BEGIN
      SUM:=0;
      FOR P:=(IF J<=NEWN-N+1 THEN 1 ELSE J ) UNTIL N DO
           SUM:=SUM+G(I,P)*S(P,J);
      TEMP(J):=SUM;
    END;
    FOR J:=1 UNTIL NEWN DO C(I,J):=TEMP(J)
  END
END
ELSE
FOR I:=1 UNTIL M(BLOCKS) DO
FOR J:=1 UNTIL NEWN DO C(I,J):=G(I,J);
FOR I:=1 UNTIL M(BLOCKS) DO
BEGIN
  FOR J:=I UNTIL K(BLOCKS) DO C(I,J+NEWN):=0;
  FOR J:=1 UNTIL (IF I>K(BLOCKS) THEN K(BLOCKS) ELSE I)DO
                C(I,J+NEWN):=L(I,J)
END;
```

.../116

```
COMMENT
  REDUCING C TO LOWER TRIANGULAR FORM.THE NUMBER OF ELEMENTS
  IN A ROW TO BE ELIMINATED IS NEWN;

SROWNO:=N;
COUNT:=1;
FOR ISTEP:=NEWN STEP - 1 UNTIL 1 DO
BEGIN
  COL:=ISTEP+1;
  FOR ROW:=1 UNTIL (IF K(BLOCKS)+COUNT-1<M(BLOCKS) THEN
           K(BLOCKS)+COUNT-1 ELSE M(BLOCKS))DO
  BEGIN
    COLMINUS1:=COL-1;
    IF ABS(C(ROW,COL))>TOL THEN
    BEGIN
      GIVENS(C(ROW,COLMINUS1),C(ROW,COL),1);
      UPDATERIGHT(COLMINUS1,COL,ROW,M(BLOCKS),C);
      ROWNO:=IF SROWNO <= 0 THEN 1 ELSE SROWNO;
      UPDATERIGHT(COLMINUS1,COL,ROWNO,N,S)
    END;
    COL:=COL+1
  END;
  COUNT:=COUNT+1;
  SROWNO:=SROWNO-1
END;

COMMENT
      REDUCING THE MATRIX C TO A FULL COLUMN MATRIX;

ROW:=COL:=1;
REDUCTION:=0;
CONTINUE:=TRUE;
WHILE (CONTINUE)DO
BEGIN
  CONTINUE:=FALSE;
  WHILE((ROW <= M(BLOCKS)) AND (ABS(C(ROW,COL)) < TOL)) DO
  BEGIN
    ROW:=ROW+1;
    COL:=COL+1
  END;
  IF ROW  = M(BLOCKS) THEN REDUCTION:=REDUCTION+1;
  IF ROW < M(BLOCKS) THEN
  BEGIN
    CONTINUE:=TRUE;
    COUNT:=1;
    FOR I:=ROW+1 UNTIL M(BLOCKS) DO
    BEGIN
      IF ABS(C(I,COL+COUNT)) > TOL THEN
      BEGIN
        GIVENS(C(I,COL+COUNT-1),C(I,COL+COUNT),1);
        UPDATERIGHT(COL+COUNT-1,COL+COUNT,I,M(BLOCKS),C);
        UPDATERIGHT(COL+COUNT-1,COL+COUNT,1,N,S)
      END;
      COUNT:=COUNT+1
    END
```

.../ 117

```
 END;
 ROW:=ROW+1
END;
WRITE('REDUCED (G,L) IN LOWER TRAPEZOIDAL FULL COLUMN RANK',
                ' MATRIX FOR BLOCK NO ',BLOCKS);
BLANKLINE(2);
PRINT(C,M(BLOCKS),N+K(BLOCKS));
BLANKLINE(2);

COMMENT
 THE RANK OF C IS M(BLOCKS)-REDUCTION.NOW IF REDUCTION > 0
 THEN SUM OF FIRST REDUCTION ELEMENTS OF (Y-G*D)SHOULD BE
 VERY SMALL AFTER C IS MADE LOWER TRIANGULAR;

BEGIN
 LONG REAL SUM;
 FOR I:=1 UNTIL M(BLOCKS)DO
 BEGIN
  SUM:=0;
  FOR J:=1 UNTIL N DO SUM:=SUM+G(I,J)*D(J);
  Y(I):=Y(I)-SUM
 END
END;

COMMENT

 REDUCING THE FULL COLUMN RANK MATRIX BY APPLYING ROTATIONS
 FROM THE LEFT TO LOWER TRIANGULAR FORM.;

FOR I:=1 UNTIL REDUCTION DO
BEGIN
 FOR ROW:=M(BLOCKS)-1 STEP -1 UNTIL I DO
 BEGIN
  COL:=ROW-I+1;
  IF ABS(C(ROW,COL)) > TOL THEN
  BEGIN
   GIVENS(C(ROW,COL),C(ROW+1,COL),0);
   UPDATELEFT(ROW,ROW+1,COL,C);
   UPDATEY(ROW,ROW+1,Y)
  END
 END
END;

COMMENT

 CHECKING THE RESIDUE WHEN THE UNDERDETERMINED SYSTEM IS NOT
 OF FULL ROW RANK.THIS WILL INDICATE THE CONSISTENCY OF THE
 MODEL BEING CONSIDERED.;

IF  REDUCTION > 0 THEN
BEGIN
 LONG REAL SUM;
 SUM:=0;
 FOR I:=1 UNTIL REDUCTION DO SUM:=SUM+Y(I)*Y(I);
 SUM:=LONGSQRT(SUM);
 IF SUM > TOL THEN
 BEGIN
```

.../118

```
      BLANKLINE(2);
      WRITE("INCONSISTENCY IN THE MODEL IN ROW BLOCKS",BLOCKS);
      BLANKLINE(20);
      GO TO STOP
     END
    END;

    COMMENT

     SOLVING THE LOWER TRIANGULAR SYSTEM CX=Y WHERE C IS [illegible]
     MATRIX OF RANK M(BLOCKS)-REDUCTION;

    W(1):=Y(REDUCTION+1)/C(REDUCTION+1,1);
    FOR I:=2 UNTIL M(BLOCKS)-REDUCTION DO
    BEGIN
     LONG REAL SUM;
     SUM:=0;
     FOR J:=1 UNTIL I-1 DO
      SUM:=SUM+C(REDUCTION+I,J)*W(J);
      W(I):=(Y(REDUCTION+I)-SUM)/C(REDUCTION+I,I)
    END;

    COMMENT

     FORMING SW WHICH IS D
                           I
                                                        ;


    FOR I:=1 UNTIL N DO
    BEGIN
     LONG REAL SUM;
     SUM:=0;
     FOR J:=1 UNTIL M(BLOCKS)-REDUCTION DO
        SUM:=SUM+S(I,J)*W(J);
     D(I):=D(I)+SUM;
     FOR J:=1 UNTIL NEWN+REDUCTION DO
        S(I,J):=S(I,J+M(BLOCKS)-REDUCTION);
     FOR J:=NEWN+REDUCTION+1 UNTIL N+MAX DO S(I,J):=0
    END;
    NEWN:=NEWN+REDUCTION;
   END;
   BLANKLINE(7);
   WRITE("THE FIRST",N,"ELEMENTS OF THE SOLUTION VECTOR");
   FOR I:=1 UNTIL N DO WRITE("     ",D(I))
  END;
  STOP:WRITE(" ")
 END MINNORM;

 COMMENT

  MAIN PROGRAM STARTS HERE.THE NUMBER OF ROW BLOCKS IN THE
  UNDERDETERMINED SYSTEM IS PASSED INTO THE PROCEDURE MINNORM
  AS PARAMETER;

 READ (ROWBLOCKS);
 WRITE(" NUMBER OF BLOCKS IN THE MATRIX F IS",ROWBLOCKS);
 MINNORM(ROWBLOCKS)
END.
```

.../119

REFERENCES

[1] AITKEN, A.C., On least squares and linear combinations of observations, Proc. R. Soc. Edinb., 55 (1934-1935), pp.42-48.

[2] BJÖRCK, Åke, A uniform numerical method for linear estimation from general Gauss-Markoff models, Proc. 1 st Symposium on Computational Statistics (COMSTAT), Vienna, 1974, pp.131-140.

[3] BUSINGER, P. and GOLUB, G.H., Linear least squares solution by Householder transformations, Numer. Math., 7 (1965), pp.269-276.

[4] de JONG, L.S., Towards a formal definition of numerical stability, Numer. Math., 28 (1977), pp.211-219.

[5] GENTLEMAN, W.M., Least squares computations by Givens transformations without square roots, J. Inst. Math. Appl., 12 (1973), pp.329-336.

[6] GILL, P.E., GOLUB, G.H., MURRAY, W. and SAUNDERS, M.A., Methods for modifying matrix factorizations, National Physical Laboratory, Report NAC 29, Dec. 1972, Teddington, England.

[7] GOLUB, G.H., Numerical methods for solving linear least squares problem, Numer. Math., 7 (1965), pp.206-216.

.../120

[8] GOLUB, G.H. and STYAN, G.P.H., Numerical computations for univariate linear models, J. Statist. Comput. Simul., 2 (1973), pp.253-274.

[9] GROVE, R.D., BOWLES, R.L. and MAYHEW, S.C., A Procedure for estimating stability and control parameters from flight test data by using maximum likelihood methods employing a real time digital system, NASA Technical Note, NASA TN D-6735, Washington D.C., May 1972.

[10] HAMMARLING, S., A note on modifications to the Givens plane rotations, J. Inst. Math. Appl., 13 (1974), pp.215-218.

[11] JOHNSTON, J., Econometric methods, 2nd Edition, McGraw-Hill, New York, 1972.

[12] KOUROUKLIS, S., Computing weighted linear least squares solutions, M.Sc. Project Report, School of Computer Science, McGill University, 1977.

[13] LAWSON, C.L. and HANSON, R.J., Solving least squares problems, Prentice-Hall, Englewood Cliffs, N.J., 1974.

[14] NAHI, N.E., Estimation theory and applications, John Wiley & Sons, New York, 1969.

.../121

[15] PAIGE, C.C., An error analysis of a method for solving matrix equations, Math. Comput., 27 (1973), pp.355-359.

[16] PAIGE, C.C., Numerical computations for some estimation problems in Engineering, Proceedings of the Ninth Interface Symposium on Computer Science and Statistics, 1976, pp.173-176.

[17] PAIGE, C.C., Computer solution and perturbation analysis of generalized linear least squares problems, (Math. Comput., submitted 1977).

[18] PAIGE, C.C., Fast numerically stable computations for generalized linear least squares problems (Siam J. Numer. Anal., submitted 1977).

[19] PETERS, G. and WILKINSON, J.H., The least squares problem and pseudo-inverses, Comput. J., 13 (1970), pp.309-316.

[20] RAO, C.R., Linear statistical inference and its applications, Chapter 4, 2nd Edition, John Wiley & Sons, New York, 1973.

[21] SAGE, A.P. and MELSA, J.L., Estimation theory with applications to communications and control, McGraw-Hill, 1971.

.../122

[22] THEIL, H., Principles of econometrics, John Wiley and Sons, New York, 1971.

[23] ZELLNER, A., An efficient method of estimating seemingly unrelated regressions and tests for aggregation bias, J. Amer. Statist. Assoc., 57 (1962), pp.348-368.