1975

A blocked orthogonalization method for nonlinear regression

Daniel C. St. Clair

A BLOCKED ORTHOGONALIZATION METHOD FOR NONLINEAR REGRESSION

by

DANIEL CLAY ST. CLAIR, 1943-

A DISSERTATION

Presented to the Faculty of the Graduate School of the

UNIVERSITY OF MISSOURI-ROLLA

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

MATHEMATICS

1975

A K Rigler Advisor	H. D. Pyron
Henry A Wiebe	O. R. Plummer
M. E. Engelhardt	Billy E Gillett

ABSTRACT

The blocked orthogonalization algorithm for nonlinear regression developed in this work results from a study of matching problems having certain identifiable characteristics with algorithms which exploit those characteristics. The new algorithm represents an extension of an earlier algorithm by D. S. Grey using a blocked orthogonalization technique proposed by R. E. von Holdt. The result is a generalization of the Grey and the Gauss-Hartley algorithms which maintains the desirable properties of these algorithms while avoiding their more serious limitations. The new algorithm was found to be quite effective for solving problems in which the parameters in the model under consideration were "naturally" grouped.

Numerous criteria for evaluating algorithm performance are used to compare results of the new algorithm with those of the Davidon-Fletcher-Powell, Levenberg-Marquardt, Gauss-Hartley, and Grey algorithms. Acceleration of the new algorithm using Cornwell's Linear Acceleration Technique is also studied. Zangwill's convergence theory establishes validity of the new algorithm for the nonquadratic case while numerical examples exhibit its robustness.

ACKNOWLEDGEMENTS

The author wishes to express his sincere appreciation to Dr. A. K. Rigler of the Computer Science Department not only for his guidance in preparing this dissertation, but also for his personal interest in his work and in his students.

The author is grateful to the Gulf Oil Corporation for its fellowship support given through the Computer Science Department at the University of Missouri-Rolla. This help certainly aided the progress made in the research reported here.

The author also wishes to express his thanks to the staff members of both the Mathematics and the Computer Science Departments at the University of Missouri-Rolla as well as to the computer center staffs at the University of Missouri-Rolla and Western Kentucky University. Their cooperation was a great asset in preparing this dissertation.

Finally, the author wishes to thank Jean who endured not only as typist but also as wife. The author is grateful to his family for their understanding and encouragement during the years in graduate school and the preparation of this dissertation.

TABLE OF CONTENTS

Table of Contents (continued)

Table of Contents (continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

I. INTRODUCTION

In the field of mathematical programming, algorithms have been developed to solve the general nonlinear unconstrained minimization problem, viz.

$$\text{minimize } g(\vec{\alpha}) \tag{1.1}$$

where $\vec{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_m]^T \varepsilon R^m$ and g is a nonlinear, scalar valued function of the α_i's.

This paper is concerned with the special subclass of nonlinear unconstrained minimization problems called nonlinear least squares. Such problems arise when one attempts to fit a set of n data points (x_i, y_i), where $i = 1,2,\ldots,n$ with the model

$$\vec{y} = \vec{f}(x;\vec{\alpha}) + \vec{\varepsilon} \tag{1.2}$$

where $\vec{\alpha}$ is the same as in (1.1), $\vec{f}$ is nonlinear in the α_i's and $\vec{\varepsilon} = [\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n]^T$ is the error vector. Fitting a set of data in the least squares sense is equivalent to solving the problem

$$\text{minimize } J(\vec{\alpha}) = \vec{h}(\vec{\alpha})^T \, \vec{h}(\vec{\alpha}) \tag{1.3}$$

where

$$\vec{h}(\vec{\alpha}) = \vec{y} - \vec{f}(x;\vec{\alpha}) \tag{1.4}$$

is called the residual. Equation (1.3) is referred to as the nonlinear least squares problem or the nonlinear regression problem and is accompanied by the fact that the number of parameters m is strictly less than the number of observation points n (often $m<<n$). In such cases, one is selecting the best solution to an overdetermined system of equations.

Since the publication of Levenberg's paper on Damped Least Squares in 1944 [1], a great deal of effort has been devoted to nonlinear regression. Initial work was in the direction of developing a universally excellent algorithm while the most recent and more realistic attitude has been to develop classes of algorithms which work well. Chapter II of this work contains descriptions of some of these successful algorithms along with descriptions of several methods which are applicable to the general nonlinear unconstrained minimization problem (1.1).

Classifying algorithms tends to imply that an effort should be made to match problems having special and identifiable characteristics with algorithms or even classes of algorithms which exploit such characteristics. While reviews of current and past literature indicate that very little has been done in this regard, it would seem that such characteristics must exist for problems arising from particular physical situations. For example, the popular variable metric algorithms have found no favor with optical designers [2]; in fact, the major lens design programs in production today use either Damped Least Squares, Grey's method [3,4], or Glatzel's method [5-7] for solving the nonlinear least squares problem.

One purpose of this study is to find characteristics of a problem which can be identified a priori so that a solution technique can be used which takes advantage of these characteristics. D. D. Walling [8] proposed an identification technique which has been modified by other authors. This strategy consists of separating those variables in the model that are purely linear from the remaining truly nonlinear variables. Various solution techniques are then applied to the linear and nonlinear sub-problems until convergence is obtained.

Motivation for approaching the separation of variables problem as is reported here resulted from attempts to fit spectral absorptivity data from a YAG laser by using a linear combination of Lorentz functions

$$y = \sum_{j=1}^{k} \frac{\alpha_{3j-2}}{\alpha_{3j-1}^2 + (\alpha_{3j} - x)^2} + \varepsilon. \qquad (1.5)$$

Grey's algorithm, abstracted from its lens design origin, has been used to solve this problem with fairly good results [9]. The present extension of this method was suggested by Grey's original title, "Aberration Theories ...," where attention is focused upon the residual itself and not upon the normal equations. The technique used is to consider the variables as being separated into "naturally" occurring groups. For example, in equation (1.5), each set of variables $\{\alpha_{3j-2}, \alpha_{3j-1}, \alpha_{3j}\}$ forms such a group. The solution technique uses Grey's algorithm in conjunction with an idea proposed by von Holdt [10] in quite a different context. The resulting technique, described in Chapter III of this paper, exploits the fact that the parameters occur in groups.

Chapter III provides detailed descriptions of both the blocked algorithm and Grey's algorithm. It is shown how von Holdt's blocked orthogonalization process was combined with Grey's algorithm to produce the new blocked algorithm. Mathematical demonstrations are included which show that the new algorithm possesses many of the desirable characteristics of Grey's original method while maintaining a faster rate of convergence. Accelerating the blocked algorithm by using Cornwell's Linear Acceleration Technique [11] not only increased the convergence rate but provided a means of guaranteeing convergence of the algorithm for the nonlinear problem.

Chapter IV contains the results of numerical investigations conducted to compare the performance of the new blocked algorithm with that of the well-known algorithms of Davidon-Fletcher-Powell [12,13], Levenberg-Marquardt [1,14], Gauss-Hartley [15], and Grey. Grey's algorithm and the blocked algorithm were studied both with and without Cornwell's modification. Algorithms were applied to problems in which the variables were naturally grouped and numerous performance indicators were recorded. The results verify the theoretical convergence predictions of Chapter III. In addition, the new algorithm converged more consistently to global minima than did the other algorithms tested! Since parameters were related to physical aspects of the model under consideration, local minima produced by the algorithms were of interest in the numerical investigations. In this respect, some local minima produced were acceptable whereas others were not. The new algorithm outperformed the others in this aspect of these tests!

Original derivation and testing of the blocked orthogonalization algorithm has been reported in an article by D. C. St. Clair and A. K. Rigler in *Proceedings of the Computer Science and Statistics Eighth Annual Symposium on the Interface* [16].

II. REVIEW OF LITERATURE

The literature records a large number of algorithms designed to solve the nonlinear least squares problem. This chapter serves as a review of the well-established algorithms while indicating recently proposed modifications of these algorithms. Algorithms cited as general methods are applicable to the general problem (1.1) as well as to the least squares problem (1.3). Least squares methods are applicable to the least squares problem only.

Several papers appear in the current literature which survey the successful algorithms in both the general and least squares classes. Recent articles in this regard include papers by Powell [17,18], Fletcher [19], and Dennis [20]. Papers by Broyden [21], Zeleznik [22], and Huang [23] have attempted to unify several successful algorithms into various generalized forms.

The common feature of all algorithms reviewed is their attempt to move from the point $\vec{\alpha}_i$ in the parameter space R^m to the point $\vec{\alpha}_{i+1} \varepsilon R^m$ by using the equation

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i + t_i \Delta\vec{\alpha}_i . \qquad (2.1)$$

The step length t_i as well as the direction vector $\Delta\vec{\alpha}_i$ are determined by the strategy used to devise the particular method under consideration. Some algorithms choose the scalar t_i by performing a one-dimensional search, such as that shown in Appendix A, along the $\Delta\vec{\alpha}$

direction. Other algorithms, such as the Levenberg-Marquardt [1,14], select the direction of search as well as the step length simultaneously. One area of current research favors omitting the step length computation by making appropriate choices for $\Delta\vec{\alpha}_i$, for example see [24-26].

The general philosophy is to produce a point $\vec{\alpha}_{i+1} \varepsilon R^m$ such that $g(\vec{\alpha}_{i+1}) < g(\vec{\alpha}_i)$. Again, algorithms vary in their approach to the problem. The algorithm of Hooke and Jeeves [27] may actually consider points where the objective function increases from time to time. Their idea was to allow the search to continue along certain "valleys" which may eventually lead to a great reduction in the value of the objective function.

A. General Methods

Many methods for solving the nonlinear unconstrained minimization problem (1.1) rely heavily on the quadratic approximation of the objective function provided by the first three terms of the Taylor series

$$g(\vec{\alpha}_i + \Delta\vec{\alpha}_i) \simeq g(\vec{\alpha}_i) + \nabla g_i^T \Delta\vec{\alpha}_i + 1/2\ \Delta\vec{\alpha}_i^T G_i \Delta\vec{\alpha}_i \qquad (2.2)$$

where

$$\nabla\vec{g}_i = [\frac{\partial g}{\partial\alpha_1}, \frac{\partial g}{\partial\alpha_2}, \ldots, \frac{\partial g}{\partial\alpha_m}]^T \qquad (2.3)$$

is the gradient vector evaluated at the point $\vec{\alpha}_i$ and G_i is the real symmetric Hessian matrix

$$G_i = [\frac{\partial^2 g}{\partial \alpha_j \partial \alpha_k}], \ 1 \le j,k \le m, \qquad (2.4)$$

evaluated at $\vec{\alpha}_i$.

The basic idea behind many of the general as well as the least squares methods is to use part or all of the terms from the truncated Taylor series (2.2) to obtain the direction vector $\Delta\vec{\alpha}_i$. Accordingly, the general algorithms reviewed in this paper are classified as: 1.) direct search methods, 2.) methods using first partial derivatives, and 3.) methods using second partial derivatives contingent on whether the algorithm in question uses no terms, the first two terms, or all three terms respectively of (2.2).

In the case of the quadratic objective function, (2.2) is an exact representation of $g(\vec{\alpha})$. Algorithms recognizing this equality converge in m or fewer steps when $g(\vec{\alpha})$ is quadratic. These algorithms are said to possess property Q. While such algorithms have an excellent rate of convergence for quadratic problems, fast convergence for the nonquadratic objective cannot be guaranteed until a satisfactory neighborhood of the solution point $\vec{\alpha}^*$ is reached.

Algorithms not possessing property Q converge to the solution in an asymptotic manner. Studies by Himmelblau

[28] indicate these algorithms converge slower than property Q algorithms; however, they are dependable when applied to a wide range of problems.

1. Direct Search Methods

The algorithm of Rosenbrock [29] as well as that of Hooke and Jeeves [27] are members of the class of direct search methods. The development of such methods has been guided by extensive experience and by thinking of the problem as that of following valleys down the side of a mountain. Hence, they developed in an intuitive sort of way. These methods ignore, for one reason or another, the information available from equation (2.2) and proceed to obtain the direction vector $\Delta\vec{\alpha}_i$ by observing values of the function at previous points. In turn, these observations are combined with a strategy for selecting the search direction at each point $\vec{\alpha}_i$. A search in the $\Delta\vec{\alpha}_i$ direction follows in order to determine the appropriate value of t_i and the resulting $\vec{\alpha}_{i+1}$. Methods in this class tend to be slower than those of the other classes, but are attractive when it is not desirable or possible to evaluate partial derivatives.

In mountain climbing terminology, Rosenbrock's method tries to follow valleys to obtain a minimum function value by attempting to identify the direction of the valley. This direction is then used as a search direction. To accomplish this objective, Rosenbrock's method searches m mutually orthogonal directions $\vec{d}_i$, one at a time. Thus, on the first iteration, the initial estimate $\vec{\alpha}_0$ is changed to $\vec{\alpha}_0 + t_1\vec{d}_1$, this in turn is changed to $\vec{\alpha}_0 + t_1\vec{d}_1 + t_2\vec{d}_2$, and so on until m iterations have replaced the initial estimate by

$$\vec{\alpha}_0 + \sum_{i=1}^{m} t_i\vec{d}_i \; .$$

After every m iterations, the set of m search directions $\vec{d}_i$ is replaced by the set

$$\vec{d}_k^* = \sum_{i=k}^{m} t_i\vec{d}_i$$

where k = 1,2,...,m. This new set of directions is orthogonalized by the familiar Gram-Schmidt orthonormalization process and the iterative process is repeated. Hence, the method aligns the first search direction along the valley which has just produced a favorable reduction in the value of the objective function. A further development of this idea has been pursued by Davies, Swann, and Campey [30].

Hooke and Jeeves' Pattern Search [27] also seeks to identify the direction of valleys, but has one additional feature not used by Rosenbrock's method. The Hooke and

Jeeves algorithm realizes that a long straight step taken along a valley may produce an increase in function value. This causes no problem since it is easy to identify the direction back down to the floor of the valley. The Hooke and Jeeves algorithm consists of making an "exploratory move" followed by a "pattern move." Suppose one is at the point $\vec{\alpha}_i$ as a result of an exploratory move made from the point $\vec{\alpha}_{i-1}$. The next move is a pattern move made by choosing $\vec{\alpha}_{i+1} = \vec{\alpha}_i + (\vec{\alpha}_i - \vec{\alpha}_{i-1})$. Next, an exploratory move is made from $\vec{\alpha}_{i+1}$ by taking small steps along each of the coordinate directions in an attempt to reduce the value of the objective function. Call the new point reached $\vec{\alpha}_{i+2}$. If $g(\vec{\alpha}_{i+2}) < g(\vec{\alpha}_i)$ a pattern move is made from $\vec{\alpha}_{i+2}$. Otherwise, one makes a new exploratory move from $\vec{\alpha}_i$.

The pattern moves can take large steps along valleys while the exploratory moves lead back down to the valley floor. The fact that $g(\vec{\alpha}_{i+1})$ is permitted to be greater than $g(\vec{\alpha}_i)$ makes the Hooke and Jeeves method particularly attractive for use in optimizing objective functions having long curved valleys.

Although the above algorithms converge asymptotically, there exist a number of direct search algorithms with property Q. Lessman's recent work [31] includes a thorough discussion of both types of nonderivative algorithms and the strategies employed in their developments.

2. Methods Using First Partial Derivatives

General methods which use the first partial derivatives of the objective function are required to provide or approximate values of the gradient vector, $\nabla\vec{g}$. Some methods in this class obtain the direction vector by approximating the objective function with two terms of the Taylor series expansion, viz.

$$g(\vec{\alpha}_i + \Delta\vec{\alpha}_i) \simeq g(\vec{\alpha}_i) + \nabla\vec{g}_i^T\Delta\vec{\alpha}_i \qquad (2.5)$$

where $\nabla\vec{g}_i$ and $\Delta\vec{\alpha}_i$ are as in (2.2) and (2.3). The approximation (2.5) is then solved for $\Delta\vec{\alpha}_i$. These methods are known as steepest descent algorithms. Other methods in this class use gradient information to estimate the Hessian matrix G_i of (2.2) and (2.4). Such methods are called quasi-Newton in contrast to Newton algorithms which require the calculation of second derivatives.

a. Steepest Descent Algorithms

One of the earliest steepest descent algorithms was developed in 1847 by Augustin Cauchy [32] for the purpose of computing stellar orbits. Referred to as the steepest descent algorithm, its main strategy lies in choosing

$$\Delta\vec{\alpha}_i = -\nabla\vec{g}_i$$

since this gives the direction of maximum decrease in function value. Powell [17] states the details concerning this strategy. The algorithm moves from the point $\vec{\alpha}_i$ to the next point $\vec{\alpha}_{i+1}$ by means of the step $\Delta\vec{\alpha}_i$, i.e.

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i + t_i \Delta\vec{\alpha}_i . \qquad (2.1)\text{bis}$$

The scalar t_i, chosen by minimizing the single variable function $g(\vec{\alpha}_i + t_i\Delta\vec{\alpha}_i)$, is necessary for nonlinear problems to prevent the algorithm's predicting a point beyond the region in which the Taylor series expansion is accurate.

Cauchy's method is well known for its sensitivity to such factors as problem scaling and initial point selection. Hence, for objective functions having high helical contours, the asymptotic convergence is slow. However, it is one of the few optimization methods supported by convergence theorems [33].

Many methods have been patterned after Cauchy's method of steepest descent which attempt to overcome the problems of slow convergence. The Supermemory Gradient Method of Cragg and Levy [34] attempts to speed convergence of Cauchy's algorithm by using information from previous moves. This technique selects the point $\vec{\alpha}_{i+1}$ by using the equation

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i + \Delta\vec{\beta}_i$$

where

$$\Delta\vec{\beta}_i = -t_i \nabla \vec{g}_i + \sum_{j=1}^{k} u_j \Delta\vec{\beta}_j .$$

The scalar k denotes the number of past iterations "remembered." The undesirable feature of this algorithm is the need to perform a multi-dimensional search in order to determine the scalars t_i and u_j.

Further attempts to speed convergence have resulted in the development of algorithms which search in conjugate directions. The following serve to explain the basic details of this strategy.

DEFINITION 2.1 The quadratic function

$$g(\vec{\alpha}) = a + \vec{c}^T\vec{\alpha} + 1/2\ \vec{\alpha}^T Q\vec{\alpha} \tag{2.6}$$

defined on R^m is said to be positive definite provided

$$\vec{\alpha}^T Q\vec{\alpha} > 0 \qquad \text{for all } \vec{\alpha} \neq 0.$$

DEFINITION 2.2 Given the positive definite quadratic function

$$g(\vec{\alpha}) = a + \vec{c}^T\vec{\alpha} + 1/2\ \vec{\alpha}^T Q\vec{\alpha},$$

defined on R^m, any two vectors $\vec{p},\vec{q} \in R^m$ are said to be conjugate with respect to Q provided they are nonzero and

$$\vec{p}^T Q\vec{q} = 0. \tag{2.7}$$

Any set of vectors $\{\vec{p}_i\} \subset R^m$ for which $\vec{p}_i^T Q\vec{p}_j = 0$ when $i \neq j$ is said to be mutually conjugate with respect to Q.

The following two theorems make conjugate direction algorithms particularly appealing. The proofs of these theorems are well-known in the literature and can be found in Appendix B.

<u>THEOREM 2.1</u> If $\{\vec{p}_i\} \subset R^m$ is a set of vectors which are mutually conjugate with respect to matrix Q of equation (2.6), then $\{\vec{p}_i\}$ is a linearly independent set.

Theorem 2.1 is not vacuous since the eigenvectors of Q form such a set of m mutually conjugate vectors. Furthermore, theorems from linear algebra guarantee that any set of m conjugate vectors span R^m.

<u>THEOREM 2.2</u> If $\{\vec{p}_i\} \subset R^m$ is a set of m mutually conjugate vectors with respect to Q, then the positive definite quadratic of equation (2.6) can be minimized by sequentially minimizing in each of the directions $\vec{p}_1, \ldots, \vec{p}_m$ exactly once.

Theorem 2.2 guarantees that an algorithm searching in conjugate directions will minimize the positive definite quadratic $g(\vec{\alpha})$ in m or fewer steps. Thus, any algorithm which develops and sequentially searches m conjugate directions will possess property Q.

In 1952, Hestenes and Stiefel [35] devised an ingenious conjugate direction scheme to accelerate the convergence of Cauchy's steepest descent method. Their idea was to begin searching in the steepest descent direction but to select the remaining search directions so that they would be conjugate to all previously searched directions. To further increase this rate of convergence, Fletcher and Reeves [36] suggested restarting the method every m + 1 steps with a search in the steepest descent direction. Luenberger [37] developed a conjugate direction method for minimizing constrained functions. In this case, matrix Q is not positive definite.

b. Quasi-Newton Algorithms

Experience and the literature indicate that the step direction $\Delta\vec{\alpha}_i$ can better be determined by algorithms which use three terms of the Taylor series instead of two; however, this implies a knowledge of second partial derivatives. Since the task of calculating second partial derivatives can be very laborious, a group of algorithms has been proposed which takes account of the second derivative terms by using only values of the first partial derivatives of the objective function. Such methods are referred to as quasi-Newton algorithms in contrast to Newton algorithms which require explicit evaluation of second derivatives.

Algorithms in this class are also referred to as variable metric algorithms. While the class contains algorithms with and without one-dimensional searches, those utilizing such searches possess property Q. Huang[23] has shown that variable metric as well as conjugate direction algorithms can be described in a generalized algorithm. Huang and Levy [38] present numerical results that show under certain conditions some of these algorithms produce identical results.

In his Projection Method, Zoutendijk [39] observed that if $\vec{p}$ is used as the direction of search in moving from $\vec{\alpha}_i$ to $\vec{\alpha}_{i+1}$, then for the quadratic objective function, equation (2.7) is equivalent to

$$(\nabla\vec{g}_{i+1} - \nabla\vec{g}_i)^T\vec{q} = 0.$$

Hence, this observation provides a means of obtaining second derivative information while using only first derivatives. Zoutendijk's algorithm is a conjugate directions method; therefore, it possesses property Q.

Consider again the case where the objective function $g(\vec{\alpha})$ is positive definite quadratic as in equation (2.6). For this rather special case, the Taylor series expansion becomes

$$g(\vec{\alpha}*) = g(\vec{\alpha}_0) + \nabla\vec{g}_0^T\Delta\vec{\alpha}_0 + 1/2\ \Delta\vec{\alpha}_0^TQ\Delta\vec{\alpha}_0 \qquad (2.8)$$

where

$$\nabla\vec{g}_0 = \vec{c} + Q\vec{\alpha}_0$$

and $\Delta\vec{\alpha}_0 = \vec{\alpha}* - \vec{\alpha}_0$. Since the objective function is quadratic, the Hessian matrix Q is a constant. Thus, equation (2.8) is an exact representation of the objective function. The step $\Delta\vec{\alpha}_0$ needed to go from the point $\vec{\alpha}_0$ to the minimum point $\vec{\alpha}*$ can be computed by differentiating equation (2.8) with respect to $\vec{\alpha}$ and then solving for $\Delta\vec{\alpha}_0$ to obtain

$$\Delta\vec{\alpha}_0 = (\vec{\alpha}* - \vec{\alpha}_0) = -Q^{-1}\nabla\vec{g}_0.$$

In other words, the step to the minimum point $\vec{\alpha}*$ is not in the direction of the negative gradient as is practiced in the steepest descent methods, but in the direction of what Wilde and Beightler [40] call a "deflected gradient." As Crockett and Chernoff [41] prove, $Q^{-1}\nabla\vec{g}$ is the gradient relative to the metric Q if the metric in R^m is defined as

$$s^2 = \vec{p}^T Q\vec{p}$$

where s^2 represents the distance between the points connected by the vector $\vec{p}$.

Thus, equation (2.1) suggests that to find the point $\vec{\alpha}_{i+1}$ when $g(\vec{\alpha})$ is not quadratic, one should compute

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i - t_i Q^{-1}\nabla\vec{g}_i$$

where t_i is determined by a one-dimensional search to insure $g(\vec{\alpha}_{i+1}) < g(\vec{\alpha}_i)$. Quasi-Newton methods avoid the task of computing and inverting Q by using gradient information from each step to approximate Q^{-1}.

One of the more successful quasi-Newton algorithms was presented by Davidon [12] and described by Fletcher and Powell [13]. The Davidon-Fletcher-Powell or DFP algorithm, as it shall be referred to in the remainder of this work, is a conjugate direction algorithm and hence, possesses property Q. The algorithm selects a sequence of positive definite matrices $\{H_i\}$ such that at the i-th step, H_i approximates Q^{-1}. The algorithm then computes

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i - t_i H_i \nabla \vec{g}_i \tag{2.9}$$

where t_i is again determined by a one-dimensional search. In choosing the H_i in such a way as to improve the rate of convergence, Davidon recommended

$$H_i = H_{i-1} + A_i + B_i \tag{2.10}$$

where H_0 is arbitrary positive definite and A_i, B_i are chosen to insure

$$Q^{-1} = \sum_{i=1}^{m} A_i$$

and

$$-H_0 = \sum_{i=1}^{m} B_i .$$

In this way, the A_i generate the true inverse of Q while the B_i remove the effects of the initial assumption H_0. Actual calculation of the A_i and B_i at the i-th step depends on the two vectors

$$\Delta\vec{\alpha}_i = \vec{\alpha}_{i+1} - \vec{\alpha}_i \tag{2.11}$$

and

$$\vec{y}_i = \nabla\vec{g}_{i+1} - \nabla\vec{g}_i . \tag{2.12}$$

At the i-th step, A_i and B_i are computed using

$$A_i = \frac{\Delta\vec{\alpha}_{i-1}\Delta\vec{\alpha}_{i-1}^T}{\Delta\vec{\alpha}_{i-1}^T\vec{y}_{i-1}} \tag{2.13}$$

$$B_i = -\frac{(H_{i-1}\vec{y}_{i-1})(H_{i-1}\vec{y}_{i-1})^T}{\vec{y}_{i-1}^T H_{i-1}\vec{y}_{i-1}} . \tag{2.14}$$

In the absence of a better alternative, the DFP algorithm selects H_0 as the identity matrix. This makes the first step of the algorithm equivalent to that of Cauchy's steepest descent method. As a matter of fact, Myers [42] shows that for a quadratic objective function, the direction vectors generated by the DFP algorithm and that of Hestenes and Stiefel [35] are scalar multiples of each other provided each takes their initial step in the steepest descent direction. Himmelblau [28] refers to the DFP algorithm as a rank two algorithm since the correction factor

$A_i + B_i$ in equation (2.10) is a matrix of rank two. A most interesting and informative description of the DFP algorithm can be found in Wilde and Beightler's Foundations of Optimization [40], under the heading "deflected gradient method."

Several interesting extensions to the DFP algorithm have been proposed. Stewart [43] has extended the algorithm so that the first partial derivatives are approximated by finite difference equations. Stewart's algorithm automatically chooses the intervals used in the finite difference approximations in such a way as to balance the truncation and roundoff errors. An Algol program of this modified version has been published by Lill [44].

McCormick and Pearson [45] recommended a reset procedure to be used with variable metric algorithms when minimizing nonquadratic functions. After m or m + 1 iterations, the algorithm is reset by selecting

$$H_{m+1} = B$$

where B is a positive definite matrix. The identity matrix is a natural choice for B. Numerical results reported by Huang and Levy [38] tend to support McCormick and Pearson's idea.

Fletcher [25] proposed a rank two algorithm with an updating process similar to the DFP. Fletcher's relation for updating H_i is based on a recursion relation for inverting matrices. In place of a one-dimensional search, a cubic interpolation with limited step length is used to determine the t_i of equation (2.9). Both Fletcher and Himmelblau [28] have tested the algorithm and report it to be about as effective as the DFP algorithm. While Fletcher's algorithm reduced the number of iterations in a given search direction, there was an increase in the number of search directions. The algorithm does not possess property Q.

Bass [24] devised a rank two algorithm with property Q which uses no one-dimensional search. Unlike the DFP algorithm which is the same at every iteration, Bass' algorithm is cyclic, i.e., it repeats every m iterations.

A class of quasi-Newton methods called rank one algorithms stemmed from the DFP algorithm. While the DFP algorithm updates H_{i-1} by using a rank two correction matrix (see equations (2.10), (2.13), and (2.14)), the methods in this class use a similar approximation but choose H_i in such a way that $(H_i - H_{i-1})$ is a symmetric matrix of rank one. Broyden [46] has shown that there is just one way of calculating H_i so that $(H_i - H_{i-1})$ is a symmetric matrix of rank one. It is given by the general formula

$$H_i = H_{i-1} - \frac{(\Delta\vec{\alpha}_{i-1}+H_{i-1}\vec{y}_{i-1})(\Delta\vec{\alpha}_{i-1}+H_{i-1}\vec{y}_{i-1})^T}{(\Delta\vec{\alpha}_{i-1}+H_{i-1}\vec{y}_{i-1})^T\vec{y}_{i-1}} \qquad (2.15)$$

where $\Delta\vec{\alpha}_{i-1}$ and $\vec{y}_{i-1}$ are defined in equations (2.11) and (2.12) respectively.

In order to use (2.15) in an algorithm, one must select a rule for calculating $\vec{\alpha}_{i+1}$ from $\vec{\alpha}_i$. The main requirement is that m applications of equation (2.15) should cause H_m to equal $-Q^{-1}$ whenever the objective function is quadratic. The large number of rules from which one may select adds to the attractiveness of this class of algorithms. In his recent article, Powell [18] presents several rank one algorithms, some of which require no one-dimensional linear searches.

3. Methods Using Second Partial Derivatives

Using all three terms of the Taylor series expansion

$$g(\vec{\alpha}_i+\Delta\vec{\alpha}_i) \approx g(\vec{\alpha}_i) + \nabla\vec{g}_i^T\Delta\vec{\alpha}_i + 1/2\ \Delta\vec{\alpha}_i^T G_i \Delta\vec{\alpha}_i \quad (2.2)\text{bis}$$

where $\nabla\vec{g}_i$ and G_i are as defined in equations (2.3) and (2.4) respectively, leads to another class of algorithms. Since equation (2.2) is exact then $g(\vec{\alpha})$ is quadratic, all algorithms in this class have property Q. Indeed, in the absence of roundoff error, convergence for the quadratic function occurs in one step. Evaluation of first and second derivatives is required to obtain the direction vector $\Delta\vec{\alpha}_i$.

One of the older and better known methods in this class is the Newton-Raphson algorithm. This algorithm estimates the position of the minimum $\vec{\alpha}^*$ by using the sufficient condition

$$\frac{\partial g(\vec{\alpha})}{\partial \vec{\alpha}} = \nabla \vec{g}_i + G_i \Delta \vec{\alpha}_i = \vec{0} \qquad (2.16)$$

where $\Delta\vec{\alpha}_i = \vec{\alpha}_{i+1} - \vec{\alpha}_i$. Solving for $\vec{\alpha}_{i+1}$ yields

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i - G_i^{-1} \nabla \vec{g}_i . \qquad (2.17)$$

For the quadratic objective, G_i is constant and equation (2.16) represents a linear system of equations. When (2.2) is not an exact representation of the objective function, equation (2.17) must be applied iteratively. As Dennis [20] notes, when $\vec{\alpha}_i$ is in a small neighborhood of $\vec{\alpha}^*$, convergence is second order, i.e.

$$||\vec{\alpha}_{i+1} - \vec{\alpha}^*|| \leq c||\vec{\alpha}_i - \vec{\alpha}^*||^2$$

where c is a constant and $\vec{\alpha}^*$ is the solution. In cases where $g(\vec{\alpha})$ is not quadratic, equation (2.2) may be a poor estimate of the objective function in the region of interest; hence, the solution of equation (2.17) may produce a value of $\vec{\alpha}$ which causes $g(\vec{\alpha})$ to increase.

To prevent the function value from increasing, equation (2.17) is replaced by

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i - t_i G_i^{-1} \nabla \vec{g}_i$$

where t_i is determined by a one-dimensional search in order to insure $g(\vec{\alpha}_{i+1}) < g(\vec{\alpha}_i)$. This modified version of the Newton-Raphson algorithm has both second order convergence in a region close to the solution and the ability to converge from very poor starting approximations. Although successful in practice, little work has been done on identifying general convergence conditions for the modified Newton-Raphson algorithm.

Perhaps the most serious disadvantages to these methods result from the requirement that both first and second derivatives must be computed. In many problems, extensive calcualtion or the unavailability of second derivatives make such methods unacceptable. This problem has motivated the development of the quasi-Newton methods.

B. Least Squares Methods

Least squares methods are motivated by attempts to solve nonlinear problems of the form

$$\text{minimize } J(\vec{\alpha}) = \vec{h}(\vec{\alpha})^T\vec{h}(\vec{\alpha}) \qquad (1.3)\text{bis}$$

where $\vec{h}(\vec{\alpha}) = [h_1(\vec{\alpha}), h_2(\vec{\alpha}), \ldots, h_m(\vec{\alpha})]^T$ is the residual defined in equation (1.4).

The basic approach to solving the least squares problem is to approximate $\vec{h}(\vec{\alpha})$ by using two terms of the Taylor series expansion about a known point $\vec{\alpha}_i$, viz.

$$\vec{h}(\vec{\alpha}_{i+1}) \simeq \vec{h}(\vec{\alpha}_i) + H\Delta\vec{\alpha}_i \qquad (2.18)$$

where

$$H = [\frac{\partial \vec{h}}{\partial \alpha_1}, \frac{\partial \vec{h}}{\partial \alpha_2}, \ldots, \frac{\partial \vec{h}}{\partial \alpha_m}] \qquad (2.19)$$

is evaluated at $\vec{\alpha}_i$. Substituting equation (2.18) into equation (1.3) and letting $\vec{h} = \vec{h}(\vec{\alpha}_i)$ yields

$$J(\vec{\alpha}_{i+1}) \simeq J(\vec{\alpha}_i) + 2\Delta\vec{\alpha}_i^T H^T \vec{h} + \Delta\vec{\alpha}_i^T H^T H \Delta\vec{\alpha}_i. \qquad (2.20)$$

The least squares algorithms to be discussed have been classified according to the strategies used in obtaining the direction vector $\Delta\vec{\alpha}_i$ from (2.20). Some methods in this class recognize that approximation (2.20) is exact whenever $\vec{h}(\vec{\alpha})$ is linear in $\vec{\alpha}$, i.e., $J(\vec{\alpha})$ is quadratic. Such least squares methods possess property Q.

Since least squares problems are a subset of the class of general nonlinear unconstrained minimization problems, all general methods can be applied to least squares problems. However, with the exception of certain pathological cases, experience indicates least squares problems are more efficiently solved by algorithms specifically designed for that purpose.

One possible reason for mediocre performance of the quasi-Newton algorithms when applied to least squares problems has been noted by Dennis [20]. Computation of the gradient and Hessian of (1.3) yields

$$\nabla J_i = 2H^T h(\vec{\alpha}_i),$$

$$G_i = 2\sum_{k=1}^{n} h_k(\vec{\alpha}_i)\nabla^2 h_k(\vec{\alpha}_i) + 2H^T H,$$

where

$$\nabla^2 h_k(\vec{\alpha}_i) = [\frac{\partial^2 h_k(\vec{\alpha}_i)}{\partial\alpha_j \partial\alpha_s}], \quad 1\leq j,s\leq m.$$

Note that $\nabla\vec{J}_i$ and G_i correspond to $\nabla\vec{g}_i$ and G_i respectively in the general Taylor series expansion (2.2). Then, computing H not only furnishes $\nabla\vec{J}_i$ but also a part of G_i. Dennis claims quasi-Newton methods are suspect for use in least squares since they attempt to build the inverse Hessian from gradient information alone. In applying such methods to least squares problems, the term $H^T H$ is thrown away--the very antithesis of the philosophy of these methods!

1. Gauss Algorithms

Karl Gauss' algorithm [47], first described in 1809, comes from the idea that when the solution $\vec{\alpha}*$ of the least squares problem has been found, then $\partial J(\vec{\alpha}*)/\partial\Delta\vec{\alpha} = \vec{0}$. Thus, differentiating equation (2.20) with respect to $\Delta\vec{\alpha}_i$ and setting the results to zero yields

$$H^T H \Delta\vec{\alpha}_i = -H^T\vec{h} = -1/2\ \nabla\vec{J}_i \qquad (2.21)$$

or

$$\Delta\vec{\alpha}_i = -(H^T H)^{-1} H^T\vec{h} \qquad (2.22)$$

where H and $\vec{h}$ are evaluated at $\vec{\alpha}_i$. After determining the step $\Delta\vec{\alpha}_i$, the new parameter estimate $\vec{\alpha}_{i+1}$ is made by computing

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i + \Delta\vec{\alpha}_i . \qquad (2.23)$$

The procedure is then repeated using $\vec{\alpha}_{i+1}$ in place of $\vec{\alpha}_i$ in equations (2.22) and (2.23). In practice, the algorithm may produce a point beyond the region in which $J(\vec{\alpha})$ can adequately be represented by the linearization of equation (2.20). In such cases, the iterates are likely to diverge.

To avoid convergence to a stationary point that is not a minimum and to insure that $J(\vec{\alpha}_{i+1}) < J(\vec{\alpha}_i)$, H. O. Hartley proposed a modification to Gauss' algorithm [15]. The Gauss-Hartley algorithm performs the process of linearization described by equation (2.20), then proceeds to solve the system of linear equations (2.21) for $\Delta\vec{\alpha}_i$. Hartley selects the new point $\vec{\alpha}_{i+1}$ using

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i + t_i \Delta\vec{\alpha}_i \qquad (2.1)\text{bis}$$

where the scalar t_i is the solution of the one-dimensional search problem

$$\text{minimize } J(t) = \text{minimize } J(\vec{\alpha}_i + t\Delta\vec{\alpha}_i)$$

for $0 \leq t \leq 1$. Both the Gauss and Gauss-Hartley algorithms possess property Q.

The Levenberg-Marquardt algorithm [1,14] is another Gauss algorithm which has been quite successful. Marquardt observed that a good choice for the direction of search lay between the negative gradient of the steepest descent algorithm and the linearization direction of Gauss' algorithm. In developing a compromise between the two algorithms, Marquardt found it possible to select both the direction of search and the step length simultaneously. Since the steepest descent process is sensitive to scaling factors, Marquardt defined a procedure for scaling the system of linear equations represented by equation (2.21). Let the scaled system be represented by

$$A^{*}\Delta\vec{\alpha}_i^{*} = \vec{f}^{*}$$

where A^{*} and $\vec{f}^{*}$ are evaluated at $\vec{\alpha}_i$. Instead of solving this equation, Marquardt solves the equation

$$(A^{*} + \lambda_i I)\Delta\vec{\alpha}_i^{*} = \vec{f}^{*} \qquad (2.24)$$

for $\Delta\vec{\alpha}_i^*$. Rescaling $\Delta\vec{\alpha}_i^*$ gives the desired $\Delta\vec{\alpha}_i$. The new solution point is given by

$$\vec{\alpha}_{i+1} = \vec{\alpha}_i + \Delta\vec{\alpha}_i. \qquad (2.23)\text{bis}$$

Marquardt observed that by choosing larger and larger values for the scalar λ_i, the direction vector $\Delta\vec{\alpha}_i^*$ rotated from the direction vector of the Gauss algorithm to the direction of Cauchy's steepest descent vector. Hence, it is always possible to select a sufficiently large λ_i such that

$$J(\vec{\alpha}_{i+1}) < J(\vec{\alpha}_i). \qquad (2.25)$$

Some form of trial and error is necessary in the selection of λ_i which will satisfy the inequality (2.25) while producing rapid convergence of the algorithm to the desired least squares result.

The Levenberg-Marquardt algorithm determines rather accurately the maximum neighborhood in which the truncated Taylor series gives an adequate representation of the nonlinear model. Hence, the algorithm avoids the problems of the Gauss algorithm while retaining that algorithm's ability to converge rapidly to the minimum value of $\vec{\alpha}$ once the vicinity of this point has been reached. Further, both the Levenberg-Marquardt and Cauchy steepest descent algorithms have the ability to converge from an initial guess $\vec{\alpha}_0$ which may be outside the region of convergence of other methods; however, the Levenberg-Marquardt algorithm avoids the slow convergence of Cauchy's method caused by poor conditioning of the $J(\vec{\alpha})$ surface. Therefore, the algorithm combines the best features of the Gauss and Cauchy algorithms while avoiding their most serious limitations.

Success of the Levenberg-Marquardt algorithm has led several authors to try modifications of the basic algorithm. Buchele [48] selects the damping factor for use in equation (2.24) by using information from the matrix of second derivatives while Tabata and Ito [49] select each damping factor based on results of the three preceding function evaluations. Brown and Dennis [50] have proposed a finite difference analog of the Levenberg-Marquardt algorithm. Meyer [51] has proposed an interesting combination of the Levenberg-Marquardt algorithm and the Gauss-Hartley algorithm. Meyer reports favorable results from his algorithm which at each step first solves the Levenberg-Marquardt problem and then does the one-dimensional search proposed by Hartley before accepting the new parameter values. A recent modification by Smith and Shanno [52] allows the damping factor of equation (2.24) to be negative as well as positive. Smith and Shanno also use the steepest descent step as an alternative in cases where the problem becomes ill-conditioned. Jones'

Spiral [53] differs from these modifications in that no matrix inversions are required.

2. Grey's Orthonormal Optimization Program

A conjugate direction algorithm for solving least squares problems arising from the design of imaging optics was developed by D. S. Grey [3,4]. Since its inception, Grey's Orthonormal Optimization Program, GOOP, has been used in optics. Although this method has found little favor in the literature, it is being used successfully at several industrial and research centers.

Feder [54] indicates that the individual elements of the residual or aberration vector $\vec{h}(\vec{\alpha})$ of equation (1.4) are of particular interest to the optical designer since they provide a measure of how sharp and distortion free a figure in the object plane of a lens system will appear in the image plane. Accordingly, Grey solved the system (2.21) in such a way as to focus attention on the aberration vector. At each step, he computed the direction vector $\Delta\vec{\alpha}_i$ using a transformation of the coordinates generated by the Gram-Schmidt orthonormalization process. This transformation, used in a stepwise procedure to move from $\vec{\alpha}_i$ to the next point $\vec{\alpha}_{i+1}$, allows for individual adjustments of the transformed parameters.

Grey's original version of the algorithm used finite difference approximations to the derivatives since analytical evaluation of derivatives was unreasonable for his problems. For purposes of algorithm comparisons, the version of GOOP presented here uses analytic derivatives.

The algorithm was described further by Pegis, Grey, Vogl, and Rigler [55] as it related to filter design. Lesnick and Rigler [9] used the algorithm in solving spectral absorptivity problems. A detailed description of GOOP was given by Broste [56] in which he applied GOOP to control problems. In addition, he proved GOOP was a conjugate direction algorithm with property Q. Cornwell [11] modified GOOP and proved that by using his modification, convergence of the algorithm was guaranteed for any $J(\vec{\alpha})$. The work reported in this thesis presents a further modification of GOOP for use in solving regression problems in which the parameters are naturally grouped by their appearance in the model being fitted. Chapter III contains a detailed description of the GOOP algorithm as well as a description of the modifications made for this work.

3. Separation of Variables Algorithms

The literature records few least squares algorithms which exploit the arrangement of the parameters within the model being fitted. Those methods now in existence tend to separate the variables in the model into two subsets; a set of purely linear variables and a set of purely nonlinear variables. Hence, to use such techniques, the model referred to in equation (1.2) must be of the form

$$\vec{y} = \sum_{i=1}^{p} \alpha_i \vec{f}_i(x,\vec{\beta}) + \varepsilon \tag{2.26}$$

where the α_i are the p linear parameters in the model and each $\vec{f}_i(x,\vec{\beta})$ is a function of one or more of the q nonlinear parameters. Note that if R^m is the parameter space, then $m = p + q$. Fitting the model in equation (2.26) in the least squares sense is equivalent to solving the problem

$$\text{minimize } J(\vec{\alpha},\vec{\beta}) = ||A(\vec{\beta})\vec{\alpha} - \vec{y}||^2 \tag{2.27}$$

over $\vec{\alpha} \varepsilon R^p$, $\vec{\beta} \varepsilon R^q$ where $||z||$ is the usual Euclidean norm and $A(\vec{\beta}) = [\vec{f}_1(x,\vec{\beta}), \vec{f}_2(x,\vec{\beta}), \ldots, \vec{f}_p(x,\vec{\beta})]$. The solution of equation (2.27) is found by solving the following two subproblems. For a given $\vec{\beta}$, first solve the linear least squares problem

$$\text{minimize } J_L(\vec{\alpha}) = ||A(\vec{\beta})\vec{\alpha} - \vec{y}||^2 \tag{2.28}$$

over $\vec{\alpha} \varepsilon R^p$. Call the solution $\vec{\alpha}(\vec{\beta})$. With this solution to the linear least squares problem, next solve the nonlinear problem

$$\text{minimize } J_N(\vec{\beta}) = ||A(\vec{\beta})\vec{\alpha}(\vec{\beta}) - \vec{y}||^2 \tag{2.29}$$

over $\vec{\beta} \in R^q$. Methods differ by how and when they solve the two subproblems in equations (2.28) and (2.29). Their common feature is that they alternate between solving the two subproblems until convergence is obtained or until the iterations are stopped.

One of the first algorithms of this type was reported by D. D. Walling [8] in 1963. For a given $\vec{\beta}$, Walling solves the linear least squares problem (2.28) by setting $\partial J_L(\vec{\alpha})/\partial\vec{\alpha} = 0$ and solving the resulting linear system of p equations in p unknowns

$$A(\vec{\beta})^T A(\vec{\beta})\vec{\alpha} = A(\vec{\beta})^T \vec{y} \tag{2.30}$$

where $A(\vec{\beta})$ is from equation (2.27). Taking the solution $\vec{\alpha}(\vec{\beta})$ of equation (2.30), Walling applies the Gauss algorithm described earlier to the nonlinear problem (2.29). He continues alternating between the two subproblems until convergence is obtained. He compares results from his algorithm with results obtained from both the Gauss and Gauss-Hartley algorithms.

In 1971, Richard Barham [57-58] proposed an algorithm similar to that of Walling.* Barham improves the convergence rate of Walling's algorithm by using either the Gauss-Hartley or the Levenberg-Marquardt algorithms to

*It is interesting to note that H. O. Hartley was at Iowa State during the course of Walling's work and at Texas A. & M. when Barham's algorithm was developed.

solve the pure nonlinear problem. Further, whenever evaluation of the objective function is required in the solution of the nonlinear problem, Barham resolves the linear problem before selecting the best correction value. Barham reports results from solving Walling's problem as well as several other problems from the literature.

More recently, Ruhe and Wedin [59] proposed a series of three related algorithms which solve (2.27) by solving first the linear and then the nonlinear subproblems. For a given estimate $\vec{\beta}$ of the nonlinear parameters, the overdetermined linear least squares problem (2.28) is solved using Householder transformations to obtain

$$\vec{\alpha}(\vec{\beta}) = A(\vec{\beta})^{+}\vec{y} \qquad (2.31)$$

where $A(\vec{\beta})^{+}$ denotes the pseudoinverse. It is at this point that the objective function is evaluated to determine if convergence has occurred.

In the absence of convergence, the solution (2.31) to the linear problem is substituted into the nonlinear problem. Equation (2.29) becomes

$$\text{minimize } J_N(\vec{\beta}) = ||[A(\vec{\beta})A(\vec{\beta})^{+} - I]\vec{y}||^2 \qquad (2.32)$$

$$= ||Q_{A(\vec{\beta})}\vec{y}||^2$$

where $Q_A = I - P_A$ and $P_A = AA^+$. The three methods presented by Ruhe and Wedin for solving the nonlinear problem make use of the separation of variables property in that at every step the $\vec{\beta}$ parameter values returned are in the submanifold

$$S_2 = \{ \begin{bmatrix} \vec{\alpha} \\ \vec{\beta} \end{bmatrix} ; \vec{\alpha} = A(\vec{\beta})^+\vec{y}\} \subset R^m.$$

Linearization is again used in a manner similar to that used in the derivation of other least squares algorithms.

Denote the residual for the general least squares problem (2.27) by

$$\vec{h}(\vec{\alpha},\vec{\beta}) = A(\vec{\beta})\vec{\alpha} - \vec{y} \tag{2.33}$$

and partition the n x m partial derivative matrix with respect to $\vec{\alpha}$ and $\vec{\beta}$ as

$$H_{\vec{\alpha},\vec{\beta}} = [A|B]$$

where

$$B = [\frac{\partial A}{\partial \beta_1}\vec{\alpha}, \frac{\partial A}{\partial \beta_2}\vec{\alpha}, \ldots, \frac{\partial A}{\partial \beta_q}\vec{\alpha}]$$

is n x q and A denotes the n x p matrix $A(\vec{\beta})$. Now in S_2, the residual becomes

$$\vec{h}(\vec{\alpha},\vec{\beta}) = \vec{h}(\vec{\beta}) = -Q_A\vec{y} \tag{2.34}$$

as indicated by equations (2.32) and (2.33). The n x q partial derivative matrix of equation (2.34) with respect to $\vec{\beta}$ is

$$H_{\vec{\beta}} = Q_A B + P_A C$$

where

$$C = A^{+T}\left(\frac{\partial A^T}{\partial \beta_1} Q_A \vec{y}, \ \ldots, \ \frac{\partial A^T}{\partial \beta_q} Q_A \vec{y}\right).$$

Using the above notations, the incremental vector $\Delta\vec{\beta}_i$ for the nonlinear problem is obtained by one of the following three algorithms

$$\text{I.} \quad \Delta\vec{\beta}_i = [Q_{A_i} B_i + P_{A_i} C_i]^+ Q_{A_i} \vec{y}$$

$$\text{II.} \quad \Delta\vec{\beta}_i = [Q_{A_i} B_i]^+ Q_{A_i} \vec{y}$$

$$\text{III.} \quad \Delta\vec{\beta}_i = B_i^+ Q_{A_i} \vec{y}.$$

Then compute

$$\vec{\beta}_{i+1} = \vec{\beta}_i + t_i \Delta\vec{\beta}_i$$

where $0 \leq t_i \leq 1$ is the steplength parameter obtained by a one-dimensional search. As with Barham's algorithm, Ruhe and Wedin indicate that whenever evaluation of the objective function is required in the one-dimensional search, the linear problem should be resolved. In the numerical examples presented, they always choose $t_i = 1$.

The following should be noted in connection with the above algorithms. Algorithm I is the Gauss algorithm applied to the restricted problem (2.32) as it was originally derived in a paper by Golub and Pereyra [60]. Algorithm II is the adaptation of the Gauss algorithm applied to the general problem (2.27) and modified to take advantage of the structure of (2.32). Algorithm III is very similar to Walling's algorithm in that minimization is first done with respect to the linear parameters and then the nonlinear parameters.

Other approaches to the separation of variables problem include work by Krough [61] and Kaufman [62]. These modifications of Golub and Peryra's algorithm tend to improve the efficiency of that algorithm. Osborne [63] used a transformation to remove the linear parameters from the model. This method requires solving a nonlinear problem having a linear constraint.

III. BLOCKED ORTHOGONALIZATION

The primary interest in this study is to match problems having characteristics which may be identified a priori with algorithms which exploit these characteristics. While the literature records algorithms which separate model parameters into groups of purely linear and purely nonlinear parameters, the algorithm reported here is applicable to any problem in which there is a "natural" parameter grouping. Such groupings are particularly appealing in solving least squares problems since parameters in these problems are directly related to the physical appearance of the data being fit. Further, since the groupings can be chosen independently of the linearities and nonlinearities of the model, the new algorithm is more general than its predecessors.

As an example of a model in which the parameters are naturally grouped, consider the problem of fitting spectral absorptivity data from a YAG laser by using a linear combination of Lorentz functions*

$$y = \sum_{j=1}^{k} \frac{\alpha_{3j-2}}{\alpha_{3j-1}^{2} + (\alpha_{3j}-x)^{2}} + \varepsilon. \qquad (1.5)\text{bis}$$

*A discussion of the origin of this problem in physical optics can be found in the text Radiation and Optics by J. M. Stone [64].

Lesnick and Rigler [9] used Grey's GOOP [3,4] to solve this problem. Mireles [65] solved a simpler version of the problem using one-dimensional coordinate searches. In both cases, parameter estimates were individually adjusted. Neither algorithm took advantage of the fact that each term in the above sum corresponds to a peak in the graph of the data. Moreover, the following estimates of peak appearance are available for the j-th peak

$$\text{amplitude} \;:\; a_{3j-2} / a_{3j-1}^2,$$

$$\text{half-width} \;:\; a_{3j-1},$$

$$\text{peak location} \;:\; a_{3j}.$$

Hence for this problem, a nonlinear least squares algorithm is desired which will take initial estimates of the parameters and adjust them in groups of three (i.e. one peak at a time). The fact that each group contains both linear and nonlinear parameters makes previous separation of parameter techniques of little value in solving this type of problem.

In order to obtain a nonlinear least squares algorithm in which model parameters can be adjusted in groups instead of separately, a block orthogonalization method proposed by R. E. von Holdt [10] in quite a different context was combined with the conjugate direction algorithm of D. S.

Grey [3,4]. While it is reasonable to approach any conjugate direction algorithm in a similar manner, the choice of Grey's method for modification was encouraged by its success with practical problems as well as the ease by which the necessary modifications could be made. The present extension was suggested by Grey's original algorithm where attention was focused upon the residual or aberration vector (1.4) and not upon the normal equations (2.21).

This chapter presents a rather detailed description of Grey's algorithm followed by a description of the proposed blocked orthogonalization algorithm. Since many of the mathematical properties of the new algorithm are simply generalizations of the properties known for Grey's method, discussion of such properties is deferred until the last of the chapter where it is shown that for the linear case, the set of parameter values produced by the new method is a subset of those produced by Grey's method. Acceleration of the blocked algorithm using Cornwell's Linear Acceleration Technique [11] is sufficient to guarantee convergence of the blocked algorithm for the nonlinear least squares problem.

A. Grey's Orthonormal Optimization Program

In Grey's Orthonormal Optimization Program (GOOP), the basic approach to solving the least squares problem is to proceed from a given parameter estimate $\vec{\alpha}_0$ to a better estimate $\vec{\alpha}$ by means of the incremental equation

$$\vec{\alpha} = \vec{\alpha}_0 + \Delta\vec{\alpha}.$$

While many strategies for selecting $\Delta\vec{\alpha}$ have been proposed, all are required to produce a value of $\vec{\alpha}$ which results in reduction of the objective function

$$J(\vec{\alpha}) = \vec{h}(\vec{\alpha})^T\vec{h}(\vec{\alpha}) \qquad (1.3)\text{bis}$$

where $\vec{h}(\vec{\alpha})$ is the residual of equation (1.4). As in the methods described in Chapter II, Grey uses linearization to replace $\vec{h}(\vec{\alpha})$ in equation (1.3) by the first two terms of its Taylor series expansion about $\vec{\alpha}_0$, viz.

$$\vec{h}(\vec{\alpha}) \simeq \vec{h} + H\Delta\vec{\alpha} \qquad (3.1)$$

where H_{ik} is the k-th element of the gradient vector evaluated at the i-th data point (see equation (2.19)) and $\vec{h} = \vec{h}(\vec{\alpha}_0)$. Recall that (3.1) is exact when $\vec{h}$ is linear.

Substituting equation (3.1) into equation (1.3) and setting $\partial J/\partial\vec{\alpha} = 0$ as was done in Gauss' method gives the linear least squares problem

$$H^T H\Delta\vec{\alpha} = -H^T\vec{h}. \qquad (3.2)$$

Grey's procedure for solving this problem uses the classical Gram-Schmidt orthonormalization process to generate a coordinate transformation from the $\Delta\vec{\alpha}$-space to a $\Delta\vec{\gamma}$-space in which the transformed version of equation (3.2) has an "easy solution." The $\Delta\vec{\gamma}$ obtained is then transformed back to obtain the $\Delta\vec{\alpha}$ which reduces J.

More specifically, matrix H in equation (3.2) is replaced by the matrix product GB, viz.

$$B^TG^TGB\Delta\vec{\alpha} = -B^TG^T\vec{h}$$

where the columns of G are orthonormal and B is the upper triangular Gram-Schmidt transformation matrix.* By simplifying and identifying $B\Delta\vec{\alpha}$ as the variable in the $\Delta\vec{\gamma}$-space, the solution of equation (3.2) in the $\Delta\vec{\gamma}$-space is reduced to

$$\Delta\vec{\gamma} = B\Delta\vec{\alpha} = -G^T\vec{h}. \tag{3.3}$$

The upper triangular form of the nonsingular matrix B is easily inverted to obtain

$$\Delta\vec{\alpha} = B^{-1}\Delta\vec{\gamma}.$$

The orthonormal transformation to the $\Delta\vec{\gamma}$-space decouples the effect of the parameters in the $\Delta\vec{\alpha}$-space on the quadratic elements of J. This coupling effect is particularly strong in the case of an extremely non-

*No relationship is being implied between this G and the Hessian G of Chapter II.

linear function. In the new space each component of $\Delta\vec{\gamma}$ can be independently adjusted to reduce J without affecting adjustments already made to previous elements. This independence is guaranteed since the partials of $\vec{h}$ with respect to each $\Delta\vec{\gamma}$ component are the orthonormal columns of G.

Details for computing the elements of G and B by the classical Gram-Schmidt orthonormalization process are given below.

$$\vec{G}_1 = \vec{H}_1/||\vec{H}_1|| \tag{3.4}$$

$$\vec{D}_i = \vec{H}_i - \sum_{k=1}^{i-1} \vec{G}_k B_{ki} \tag{3.5}$$

$$\vec{G}_i = \vec{D}_i/||\vec{D}_i|| \qquad i=2,3,\ldots,m \tag{3.6}$$

where

$$B_{ki} = \vec{G}_k^T \vec{H}_i, \tag{3.7}$$

$$||\vec{D}_i|| = (\vec{D}_i^T \vec{D}_i)^{1/2}, \tag{3.8}$$

and $\vec{H}_i$ denotes the i-th column of H.

An important feature of the above transformation is that the solution of equation (3.2) given by equation (3.3) can be computed in a stepwise manner allowing use of the most current parameter estimates in adjusting the $\Delta\vec{\gamma}$ components. In other words, the i-th component of $\Delta\vec{\gamma}$ is given by

$$\Delta\gamma_i = -\vec{G}_i^T\vec{h}. \qquad (3.9)$$

Computing $\vec{C}_i$, the i-th column of B^{-1} at each step would allow determination of the appropriate change in the $\Delta\vec{\alpha}$-space, i.e.

$$\Delta\vec{\alpha}_i = \vec{C}_i\Delta\gamma_i.$$

Observing that the orthonormalization process builds the B matrix one column at each step, B can be constructed as the product of m matrices each of the form

$$B_i = \begin{bmatrix} 1 & & & B_{1i} & & & \cdot\cdot 0 \\ & \cdot & & B_{2i} & & & \vdots \\ & & \cdot & \cdot & & & \\ & & & B_{ii} & & & \\ & & & & \cdot & & \\ & & & & & \cdot & \\ \vdots & & & & & & \cdot \\ 0\cdot\cdot & & & & & & 1 \end{bmatrix}$$

where the elements in the matrix are computed by equation (3.7). Note that $B_{ii} = ||\vec{D}_i||$. If B(i) represents that part of the matrix B constructed at the i-th step, then

$$B(i) = B_iB_{i-1}\cdots B_1 \qquad (3.10)$$

and

$$B = B(m).$$

Computing $B(i)^{-1}$ would give the part of B^{-1} which has been constructed at the i-th step. Applying simple matrix algebra to equation (3.10) gives

$$B(i)^{-1} = B_1^{-1} \ldots B_{i-1}^{-1} B_i^{-1} = B(i-1)^{-1} B_i^{-1} \qquad (3.11)$$

and

$$B^{-1} = B(m)^{-1}.$$

Equation (3.11) is particularly appealing since

$$B_i^{-1} = \begin{bmatrix} 1 & & & -B_{1i}/B_{ii} & & \cdot\cdot\, 0 \\ & \cdot & & -B_{2i}/B_{ii} & & \vdots \\ & & \cdot & \vdots & & \\ & & & 1/B_{ii} & & \\ & & & & \cdot & \\ \vdots & & & & & \cdot \\ 0\,\cdot\cdot & & & & & 1 \end{bmatrix}. \qquad (3.12)$$

If $B(i)^{-1}$ is computed at each step, only the elements of the i-th column of B and the matrix $B(i-1)^{-1}$ are required. Since future steps do not alter the i-th column of $B(i)^{-1}$, it is equal to $\vec{C}_i$, the i-th column of B^{-1}.

The iteration scheme for GOOP can be summarized as follows.

Initially choose $\vec{\alpha}_0$ and evaluate $\vec{h}(\vec{\alpha}_0) = \vec{y} - \vec{f}(x;\vec{\alpha}_0)$

At the i-th step compute

(a) $\vec{H}_i = \partial\vec{h}/\partial\alpha_i$: i-th column of H.

(b) $\vec{G}_i$: i-th column of G from the $\Delta\vec{\gamma}$-space from equations (3.4-3.8).

(c) $\vec{C}_i$: the i-th column of B^{-1} from equations (3.11, 3.12).

(d) $\Delta\gamma_i$: the scalar from equation (3.9) to minimize $\vec{h}(\vec{\alpha})$ in the $\vec{C}_i$ direction.

(e) $\vec{\alpha}_i = \vec{\alpha}_{i-1} + \vec{C}_i\Delta\gamma_i$: update.

Broste [56] has shown that GOOP is a conjugate direction algorithm. Therefore GOOP has property Q and m or fewer cycles of the algorithm will minimize an objective that is a pure linear least squares function. When $\vec{h}(\vec{\alpha})$ is nonlinear, the algorithm may produce a point lying outside the region in which the linearization assumption (3.1) is accurate. In such cases, the point chosen by the algorithm may actually increase the value of the objective function.

Broste described several modifications of the above iteration scheme when Grey's algorithm is being applied to a nonlinear residual. These modifications prevent the algorithm from producing points beyond the region accurately described by the linearization while maintaining the fastest possible rate of convergence.

The first modification is to replace step (e) in the above iterative scheme by

$$(e) \quad \vec{\alpha}_i = \vec{\alpha}_{i-1} + t_i \vec{C}_i \Delta\gamma_i .$$

Broste determined the scalar t_i by using a series of scaling estimates. The work presented here uses a one-dimensional search in the $\vec{C}_i \Delta\gamma_i$ direction to determine a value of t_i such that $J(\vec{\alpha}_i) < J(\vec{\alpha}_{i-1})$.

Further, since H is not a constant matrix in the nonlinear case, $\vec{H}_i$ should be re-evaluated at the i-th step using the most recent parameter estimates. In like manner, the most recent value of the residual at $\vec{\alpha}_{i-1}$ should be used in determining $\Delta\gamma_i$. Since the residual has already been evaluated as a result of the one-dimensional search, no new computation is needed.

In the case of the nonlinear residual, GOOP becomes an infinite process. Convergence is not guaranteed in m steps as it was for the linear problem. For this reason, it is necessary to continue iterating until the value of the objective function becomes sufficiently small or until no further reduction of the objective function is possible.

Cornwell [11] has shown that it is worthwhile to insert a spacer step at the end of every m steps. A spacer step is a procedure which given a nonsolution point $\vec{\alpha}_{i-1}$ can find an $\vec{\alpha}_i$ such that $g(\vec{\alpha}_i) < g(\vec{\alpha}_{i-1})$. Beginning at $\vec{\alpha}_0$ Cornwell applies GOOP to the nonlinear problem for m steps to obtain the point $\vec{\alpha}_m$. He then applies his spacer step LAT, i.e. Linear Acceleration Technique, which performs a one-dimensional search in the $\vec{\alpha}_m - \vec{\alpha}_0$ direction to find a point $\vec{\alpha}_0^{(1)}$ such that $J(\vec{\alpha}_0^{(1)}) < J(\vec{\alpha}_m)$. GOOP is then restarted from the point $\vec{\alpha}_0^{(1)}$. This is not only a practical acceleration trick but a theoretical safeguard since by using Zangwill's [66] mathematical framework, Cornwell has been able to prove that modifying GOOP with LAT will guarantee convergence of the algorithm in the case of a nonlinear residual. A flow chart for LAT can be found in Appendix C.

B. Blocked Orthogonalization

Recognizing as in the YAG example that the parameters in least squares problems are often naturally grouped, it seems reasonable to seek an algorithm which adjusts parameters in groups instead of adjusting them one at a time as is done in GOOP. Further, since each group may contain both linear and nonlinear parameters, the algorithm should be capable of handling mixed groups of parameters. The algorithm proposed in this section meets these requirements. It represents a combination of Grey's algorithm with a block Gram-Schmidt process proposed by R. E. von Holdt [10] in 1962. Von Holdt originally used the blocked Gram-Schmidt process in solving discrete analogs of elliptic boundary value problems.

The new algorithm replaces the Gram-Schmidt process of equations (3.4-3.8) from GOOP with the following block Gram-Schmidt process of von Holdt.

$$G_1 = H_1 L_1 \tag{3.13}$$

$$D_i = H_i - \sum_{k=1}^{i-1} G_k B_{ki} \tag{3.14}$$

$$G_i = D_i L_i \qquad i = 2,3,\ldots,j \tag{3.15}$$

where

$$B_{ki} = G_k^T H_i \tag{3.16}$$

and the inverse of L_i is the non-singular, upper triangular matrix formed by the Cholesky decomposition of $D_i^T D_i$. It should be noted that the G_i and D_i are now nxj column matrices where j represents the number of parameters in each block. The matrices B_{ki} and L_i are jxj. A description of the Cholesky decomposition technique can be found in Appendix D while the question of the existence of L_i is discussed later in this chapter. Appendix E verifies that for a constant matrix H, the orthonormal matrix produced by the Gram-Schmidt process is identical to that produced by the blocked Gram-Schmidt algorithm!

Using the blocked Gram-Schmidt process to transform the parameter space decouples the effects of these parameters in the $\Delta\vec{\alpha}$-space on the quadratic elements of J, but it decouples them in blocks of j parameters each. Thus, j parameters can be adjusted at a time in the $\Delta\vec{\gamma}$-space without affecting adjustments already made to previous blocks. Thus, equation (3.9) becomes

$$\Delta\vec{\gamma}_i = -G_i^T\vec{h} \tag{3.17}$$

where $\Delta\vec{\gamma}$ is a j element column vector.

Constructing C_i, the i-th column matrix of B^{-1} at each step would allow one to determine the appropriate change in the $\Delta\vec{\alpha}$-space in essentially the same manner as was done before, i.e.

$$\Delta\vec{\alpha}_i = C_i\Delta\vec{\gamma}_i. \tag{3.18}$$

The matrix C_i is now an mxj matrix.

Taking advantage of the fact that the blocked Gram-Schmidt process builds matrix B in blocks of j columns at a time, matrix B can be constructed in blocks of j vectors each by taking

$$B_i = \begin{bmatrix} I & & & B_{1i} & & & \cdot\cdot 0 \\ & \cdot & & B_{2i} & & & \vdots \\ & & \cdot & \cdot & & & \\ & & & \cdot & & & \\ & & & L_i^{-1} & & & \\ & & & & \cdot & & \\ \vdots & & & & & \cdot & \\ 0 \cdot\cdot & & & & & & I \end{bmatrix}$$

where the elements of B_i are jxj matrices from the blocked Gram-Schmidt process. The corresponding inverse is

$$B_i^{-1} = \begin{bmatrix} I & & & -B_{1i}L_i & & & \cdot\cdot 0 \\ & \cdot & & -B_{2i}L_i & & & \vdots \\ & & \cdot & \cdot & & & \\ & & & \cdot & & & \\ & & & L_i & & & \\ & & & & \cdot & & \\ & & & & & \cdot & \\ & & & & & & \cdot \\ \vdots & & & & & & \\ 0 \cdot\cdot & & & & & & I \end{bmatrix} .$$

Again, letting B(i) be that part of the matrix G constructed at the i-th step one has, as with GOOP,

$$B(i)^{-1} = B(i-1)^{-1}B_i^{-1} \; . \tag{3.20}$$

Therefore, the i-th column block of $B(i)^{-1}$ will be C_i, the i-th column block of B^{-1}.

The iteration scheme for the blocked orthogonalization method appears similar to that for GOOP.

Initially choose $\vec{\alpha}_0$ and j.

Evaluate $\vec{h}(\vec{\alpha}_0) = \vec{y} - \vec{f}(x;\vec{\alpha}_0)$.

At the i-th step compute

(a) $H_i = [\partial\vec{h}/\partial\alpha_{ij-j+1} \ldots \partial\vec{h}/\partial\alpha_{ij}]$: i-th column block (an nxj matrix) of H.

(b) G_i : i-th column block (an nxj matrix) of G using equations (3.13-3.16).

(c) C_i : i-th column block (an mxj matrix) of B^{-1} computed by equations (3.19, 3.20).

(d) $\Delta\vec{\gamma}_i$: a j element column vector computed by equation (3.17).

(e) $\vec{\alpha}_i = \vec{\alpha}_{i-1} + C_i\Delta\vec{\gamma}_i$: update.

The number of parameters in each block, i.e. j, may be determined by the appearance of the parameters in the function being fitted. For the YAG problem, j is equal to 3. While the proposed algorithm blocks parameters into groups of equal size, the algorithm is easily modified to handle blocks of various sizes within the problem. For cases in which j=1, the algorithm reduces to Grey's original method. When all parameters are placed in the same block, the method becomes an implementation of the Gauss-Hartley algorithm [15].

As with GOOP, the blocked orthogonalization algorithm is applied to problems having nonlinear residuals by making the following modifications to the above procedure.

a.) Before updating at step (e), perform a one-dimensional search to determine the length of the step to be taken in the $C_i\Delta\vec{\gamma}_i$ direction.

2.) Use the most recent parameter estimates available when computing H_i and $\Delta\vec{\gamma}_i$.

3.) If convergence has not been obtained after m/j steps (i.e. one pass through the matrix of partials), set $\vec{\alpha}_0 = \vec{\alpha}_{m/j}$ and repeat the entire process. Continue until $J(\vec{\alpha})$ becomes sufficiently small or until no further reduction of $J(\vec{\alpha})$ is possible. (3.21)

Every m/j steps, Cornwell's LAT [11] can be applied to the blocked orthogonalization method in precisely the same way as it is used in Grey's algorithm provided that all the variables have not been grouped into one block. Such a grouping would turn the new algorithm into an implementation of the Gauss-Hartley method. In this case, LAT would attempt a one-dimensional search in the $\vec{\alpha}_1 - \vec{\alpha}_0$ direction, the direction which was just searched in determining $\vec{\alpha}_1$! As the results of the next chapter will verify, Cornwell's modification does accelerate the blocked orthogonalization method. As with GOOP, the LAT modification is sufficient to guarantee convergence of the blocked algorithm.

C. Mathematical Considerations

Since the blocked orthogonalization method just presented is a generalization of Grey's method, discussion of the mathematical properties of both methods has been deferred until this time. This section generalizes some of Broste's results concerning Grey's algorithm and shows that they are also true for the blocked algorithm. It shall be demonstrated that for the linear least squares problem, the set of points produced by the new method is a subset of the set of points produced by Grey's method. Unless stated to the contrary, the results obtained for blocked orthogonalization are applicable to Grey's original method.

1. Conjugate Directions

Broste has shown that the directions $\Delta\vec{\alpha}_i = \vec{C}_i\Delta\gamma_i$, $i = 1,2,\ldots,m$, generated by Grey's method are H^TH-conjugate, i.e.

$$\Delta\vec{\alpha}_i^T H^T H \Delta\vec{\alpha}_j = 0. \quad i \neq j$$

The same results can be shown for the blocked orthogonalization method of j parameters per block. Recalling that

$$\Delta\vec{\alpha}_i = C_i\Delta\vec{\gamma}_i \qquad (3.18)\text{bis}$$

and taking $i \neq j$, one obtains

$$\begin{aligned}\Delta\vec{\alpha}_i^T H^T H \Delta\vec{\alpha}_k &= \Delta\vec{\gamma}_i^T C_i^T H^T H C_k \Delta\vec{\gamma}_k \\ &= \Delta\vec{\gamma}_i^T C_i^T B^T G^T G B C_k \Delta\vec{\gamma}_k \\ &= \Delta\vec{\gamma}_i^T C_i^T B^T B C_k \Delta\vec{\gamma}_k \\ &= \Delta\vec{\gamma}_i^T E_i^T E_k \Delta\vec{\gamma}_k \\ &= 0\end{aligned}$$

since

$$BC_k = E_k = [0,0,\ldots,I,\ldots,0]^T. \qquad (3.22)$$

Each element of BC_k is a jxj matrix. Hence, the directions produced by the blocked method are H^TH-conjugate. Taking j = 1 gives Broste's proof for Grey's method.

2. Redundant Parameters

The blocked orthogonalization process depends heavily on the existence of a set of upper triangular matrices $\{L_i\}$, $i = 1,\ldots,m$. Recall that each L_i is computed by first computing

$$D_i^T D_i = (L_i^{-1})^T L_i^{-1}$$

where L_i^{-1} is the upper triangular matrix produced by the Cholesky decomposition of $D_i^T D_i$ (see Appendix D). The inversion of L_i^{-1} gives the desired L_i.

While the upper triangular form of L_i^{-1} makes matrix inversion easy, it is necessary and instructive to question the existence of L_i. Theorem 3.1 not only shows that L_i can fail to exist, but that such failure may be an indicator of poor model formulation. The following definition and lemma are necessary for use in Theorem 3.1.

DEFINITION 3.1 The function $J(\vec{\alpha})$ is said to contain redundant parameters if some model parameters are linear combinations of others.

LEMMA 3.1 If P is a symmetric positive definite matrix, then P may be decomposed by Cholesky decomposition into $P = U^T U$ where U is a nonsingular, upper triangular matrix.

Proof of Lemma 3.1 can be found in the book by F. D. Murnaghan [67].

THEOREM 3.1 Let L_i^{-1} be the upper triangular matrix produced at the i-th step of the blocked orthogonalization algorithm by the Cholesky decomposition of $D_i^T D_i$, i.e.

$$D_i^T D_i = (L_i^{-1})^T L_i^{-1}.$$

Assume L_k the inverse of L_k^{-1} exists for k = 1,2,...,i-1. Then, L_i, the inverse of L_i^{-1} exists if and only if none of the parameters in the first i blocks of parameters of $J(\vec{\alpha})$ are redundant.

PROOF It will first be shown that if L_k, k = 1,2,...,i, exist then the parameters in the first i blocks are not redundant. At the k-th step, let H(k), G(k), and B(k) represent the parts of H, G, and B respectively which have been computed by the blocked orthogonalization algorithm. In particular, B(i) has the form

$$B(i) = \begin{bmatrix} L_1^{-1} & B_{12} & \cdots B_{1i} \\ & L_2^{-1} & \vdots \\ \vdots & \ddots & \\ 0 \cdots & & L_i^{-1} \end{bmatrix} .$$

Since $B(i)$ and L_j^{-1}, $j = 1,\ldots,k$ are upper triangular matrices, the determinant

$$|B(i)| = \prod_{j=1}^{i} |L_j^{-1}| \neq 0$$

because each L_j^{-1}, $j = 1,2,\ldots,i$ is nonsingular. Now, $G(i)^T G(i) = I$ by the blocked Gram-Schmidt process so that

$$|H(i)^T H(i)| = |B(i)^T G(i)^T G(i) B(i)| = |B(i)^T B(i)| \neq 0.$$

Therefore, the columns of $H(i)$ are linearly independent and by Definition 3.1, none of the parameters in any of the first i blocks are redundant.

While the above argument is reversible, it is instructive to prove the remaining part of the theorem using the contrapositive. Assume that L_k, $k = 1,\ldots,i-1$ exist but that L_i^{-1} is singular. In particular, suppose that the p-th diagonal element of L_i^{-1} is zero. As Appendix D shows, for any given row, the Cholesky decomposition forms elements to the right of the diagonal element by dividing by that diagonal element. Hence, if the p-th diagonal element is zero the Cholesky decomposition fails and $D_i^T D_i$ cannot be decomposed in this manner. Applying Lemma 3.1 guarantees that $D_i^T D_i$ is not positive definite. Hence $|D_i^T D_i| = 0$ and the columns of D_i are linearly dependent. Since D_i is produced by a linear combination of elements from $H(i)$, it must be the case that at least one column of H(i) is a linear combination of other columns of H(i)! Therefore, the first i blocks contain redundant parameters.

Q.E.D.

Theorem 3.1 shows that the blocked algorithm possesses the ability to indicate whether or not model parameters are redundant. Further, failure of the algorithm at the i-th step due to redundancy indicates at least one parameter in the i-th block is a linear combination of other parameters in the first i blocks. The second part of the proof of this theorem indicates the computational effects produced by redundant parameters. When redundant parameters are detected, the model under consideration should be carefully re-evaluated.

3. Linear Least Squares Objective Function

While concern is primarily with the nonlinear least squares problem, some remarks about the new algorithm and the linear least squares problem are in order. Hence, the remarks made in this section are concerned with the cases in which the blocked orthogonalization algorithm is being applied to the linear least squares problem. It shall not only be shown that the new algorithm possesses property Q, i.e. the ability to converge in m or fewer steps, but the stronger result that k steps of the blocked orthogonalization algorithm theoretically produce the same result as jk steps of Grey's algorithm.

First, it shall be shown that modifications 1.) and 2.) from (3.21) are of no theoretical consequence when applying the blocked orthogonalization method to linear least squares problems. Performing a one-dimensional search in the $C_i\Delta\vec{\gamma}$ direction in step (e) of the iteration process for the blocked orthogonalization algorithm is equivalent to replacing this step by

$$\text{(e)} \quad \vec{\alpha}_i = \vec{\alpha}_{i-1} + t_i C_i \Delta\vec{\gamma}_i \qquad (3.23)$$

where t_i is the scalar determined by the one-dimensional search. Consider the following lemma.

<u>LEMMA 3.2</u> If the blocked orthogonalization algorithm is applied to the linear least squares function $J(\vec{\alpha})$, then choosing $t_i = 1$ in equation (3.23) will produce the maximum decrease in J when moving from $\vec{\alpha}_{i-1}$ to $\vec{\alpha}_i$.

<u>PROOF</u> Let ΔJ_i denote the change in $J(\vec{\alpha})$ when moving from the point $\vec{\alpha}_{i-1}$ to $\vec{\alpha}_i$ by $\vec{\alpha}_i = \vec{\alpha}_{i-1} + \Delta\vec{\alpha}_i$ where $\Delta\vec{\alpha}_i = -t_i C_i \Delta\vec{\gamma}_i$ from equation (3.23). Then,

$$\begin{aligned}\Delta J_i &= J(\vec{\alpha}_i) - J(\vec{\alpha}_{i-1}) = \vec{h}(\vec{\alpha}_i)^T\vec{h}(\vec{\alpha}_i) - \vec{h}(\vec{\alpha}_{i-1})^T\vec{h}(\vec{\alpha}_{i-1}) \\ &= [\vec{h}(\vec{\alpha}_{i-1}) + H\Delta\vec{\alpha}_i]^T[\vec{h}(\vec{\alpha}_{i-1}) + H\Delta\vec{\alpha}_i] - \vec{h}(\vec{\alpha}_{i-1})^T\vec{h}(\vec{\alpha}_{i-1}) \\ &= \Delta\vec{\alpha}_i^T H^T H\Delta\vec{\alpha}_i + 2\Delta\vec{\alpha}_i^T H^T\vec{h}(\vec{\alpha}_{i-1}).\end{aligned}$$

Since $H = GB$, $\Delta\vec{\gamma}_i = -G_i^T\vec{h}(\vec{\alpha}_{i-1})$, and $H\Delta\vec{\alpha}_i = t_iG_i\Delta\vec{\gamma}_i$ substitution yields

$$\Delta J_i = t_i^2\Delta\vec{\gamma}_i^TG_i^TG_i\Delta\vec{\gamma}_i - 2t_i\Delta\vec{\gamma}_i^TG_i^T\vec{h}(\vec{\alpha}_{i-1})$$

$$= t_i^2||\Delta\vec{\gamma}_i||^2 - 2t_i||\Delta\vec{\gamma}_i||^2.$$

Setting $\partial\Delta J_i/\partial t_i = 0$ gives the desired value of t_i,

$$\partial\Delta J_i/\partial t_i = 2t_i||\Delta\vec{\gamma}_i||^2 - 2||\Delta\vec{\gamma}_i||^2 = 0.$$

Hence, $t_i = 1$ produces the greatest decrease in J when going from $\vec{\alpha}_{i-1}$ to $\vec{\alpha}_i$.

Q.E.D.

Therefore, when $J(\vec{\alpha})$ is a linear least squares objective, no improvement in the algorithm is gained by making the one-dimensional search at each step.

Further, using the most recent parameter estimates available when computing H_i and $\Delta\vec{\gamma}_i$ at each step do not improve the convergence rate of the blocked orthogonalization algorithm. More specifically, when the residual $\vec{h}(\vec{\alpha}) = \vec{y} - \vec{f}(x;\vec{\alpha})$ is linear, the matrix H is a constant matrix and not a function of the parameters being estimated. Hence, using the most recent parameter estimates available has no effect on computing H_i. As can be seen in the proof of Lemma 3.3, it is not necessary to use the most recent parameter estimates available when evaluating $\Delta\vec{\gamma}_i$.

LEMMA 3.3 When applying the blocked orthogonalization algorithm to the linear least squares function $J(\vec{\alpha})$, the convergence rate of the algorithm is not improved by re-evaluating the residual $\vec{h}(\vec{\alpha})$ at each step of the procedure.

PROOF The only use of the residual $\vec{h}(\vec{\alpha})$ in the blocked orthogonalization algorithm occurs in the evaluation of $\Delta\vec{\gamma}_i = -G_i^T\vec{h}(\vec{\alpha}_{i-1})$. Then, it must be shown that the $\Delta\vec{\gamma}_i$ obtained at the i-th step as a result of re-evaluating the residual at each of the preceding steps is the same $\Delta\vec{\gamma}_i$ obtained if the residual is not re-evaluated.

Hence, at the i-th step

$$\Delta\vec{\gamma}_i = -G_i^T\vec{h}(\vec{\alpha}_{i-1}) = -G_i^T[\vec{y} - \vec{f}(x;\vec{\alpha}_0) - H(\vec{\alpha}_{i-1} - \vec{\alpha}_0)]$$

$$= -G_i^T[\vec{h}(\vec{\alpha}_0) - GB\sum_{k=1}^{i-1}(\alpha_k - \alpha_{k-1})]$$

$$= -G_i^T[\vec{h}(\vec{\alpha}_0) - GB\sum_{k=1}^{i-1}C_k\Delta\vec{\gamma}_k]$$

$$= -G_i^T[\vec{h}(\vec{\alpha}_0) - G\sum_{k=1}^{i-1}E_k\Delta\vec{\gamma}_k]$$

$$= -G_i^T[\vec{h}(\vec{\alpha}_0) - \sum_{k=1}^{i-1}G_k\Delta\vec{\gamma}_k]$$

$$\Delta\vec{\gamma}_i = -G_i^T\vec{h}(\vec{\alpha}_0) - \sum_{k=1}^{i-1} G_i^T G_k \Delta\vec{\gamma}_k$$

where $E_k = [0,0,\ldots,I,\ldots,0]^T$ as in equation (3.22). Since the G_k are orthogonal

$$\Delta\vec{\gamma}_i = -G_i^T\vec{h}(\vec{\alpha}_0)$$

which is the same result obtained were the residual not updated at each step.

Q.E.D.

The proofs presented above are generalizations of those given by Broste in his work on Grey's algorithm. Indeed, taking the number of parameters in each block as one, i.e. $j = 1$, will essentially give Broste's proofs. The above lemmas are important since they assure that applying the blocked orthogonalization algorithm to a linear least squares problem mathematically produces the same results, whether or not the modifications (3.21) are used. Not only can one consider solving the linear least squares problem without the modifications for the nonlinear problem, but the lemmas imply that the inner iterative process is unnecessary for the pure linear problem. The problem could be solved in one step by constructing $\vec{\alpha}* = \vec{\alpha}_0 - B^{-1}G^T\vec{h}(\vec{\alpha}_0)$.

The following theorem shows that for the linear least squares problem, the set of points generated by the blocked orthogonalization process is a subset of the set of points generated by Grey's method. Since Lemmas 3.2 and 3.3 indicate the modifications in (3.21) have no effect when either of the algorithms are applied to the linear least squares problem, the theorem is proven for unmodified algorithms.

THEOREM 3.2 Let $J(\vec{\alpha})$ be a linear least squares problem. Starting at the point $\vec{\alpha}_0$ let $\{\vec{\alpha}_k\}$, $k = 1,2,\ldots,m$ be the sequence of points generated in m-steps of Grey's algorithm and let $\{\vec{\beta}_i\}$, $i = 1,2,\ldots,m/j$ be the sequence of points generated in m/j steps of the blocked orthogonalization algorithm where j parameters have been placed in each block. Then

$$\vec{\beta}_i = \vec{\alpha}_{ij}$$

for $i = 1,2,\ldots,m/j$.

PROOF Let H = GB = QR where G and Q are the orthonormal matrices from the classical and blocked Gram-Schmidt methods respectively. B and R are the corresponding upper triangular Gram-Schmidt transformation matrices. Since H, the matrix of partial derivatives from equation (3.2), is constant in the linear least squares problem, the results of Appendix E guarantee G = Q and B = R.

From the blocked Gram-Schmidt process, let $\vec{Q}_{ki}$, k = 1,2,...,j be the k-th vector in Q_i, the i-th column block of Q and $\vec{S}_{ki}$, k = 1,2,...,j be the k-th vector in S_i, the i-th column block of R^{-1}. Note that

$$\vec{Q}_{ki} = \vec{G}_{(i-1)j+k}$$

$$\vec{S}_{ki} = \vec{C}_{(i-1)j+k}$$

where i = 1,2,...,m/j and k = 1,2,...,j. $\vec{G}_i$ is the i-th column of G and $\vec{C}_i$ is the i-th column of B^{-1} from the Gram-Schmidt process.

Let $\Delta\vec{\sigma}_k$ denote the change in the orthogonal space produced by equation (3.17) from the blocked method. Let $\Delta\gamma_k$ denote the change in the orthogonal space produced by the k-th step of Grey's method(equation (3.9)). Denote $\vec{h}(\vec{\alpha}_0)$ by $\vec{h}$.

Letting $\{\vec{\beta}_k\}$, k = 1,2,...,m/j and $\{\vec{\alpha}_i\}$, i = 1,2,...,m be the points in the statement of the theorem, proceed by induction.

For k = 1

$$\vec{\beta} = \vec{\alpha}_0 + S_1\Delta\vec{\sigma}_1 = \vec{\alpha}_0 - S_1Q_1^T\vec{h}$$

$$= \vec{\alpha}_0 - [\vec{S}_{11},\vec{S}_{21},\ldots,\vec{S}_{j1}][\vec{Q}_{11}^T\vec{h},\vec{Q}_{21}^T\vec{h},\ldots,\vec{Q}_{j1}^T\vec{h}]^T$$

Substitution yields $\vec{Q}_{k1}^T\vec{h} = \vec{G}_k^T\vec{h} = \Delta\gamma_k$ for k = 1,2,...,j. In other words, the k-th element of the vector $\vec{Q}_1^T\vec{h}$ is the change generated at the k-th step in the orthogonal space of Grey's method. Recalling $\vec{S}_{k1} = \vec{C}_k$ gives

$$\begin{aligned}\vec{\beta}_1 &= \vec{\alpha}_0 - [\vec{C}_1,\vec{C}_2,\ldots,\vec{C}_j][\Delta\gamma_1,\Delta\gamma_2,\ldots,\Delta\gamma_j]^T \\ &= \vec{\alpha}_0 - \Delta\gamma_1\vec{C}_1 - \Delta\gamma_2\vec{C}_2 - \ldots - \Delta\gamma_j\vec{C}_j \\ &= \vec{\alpha}_j .\end{aligned}$$

Now, assume the theorem holds for $i = q$, i.e. $\vec{\beta}_q = \vec{\alpha}_{qj}$. It must be shown that $\vec{\beta}_{q+1} = \vec{\alpha}_{(q+1)j}$.

$$\begin{aligned}\vec{\beta}_{q+1} &= \vec{\beta}_q + S_q\Delta\vec{\sigma}_q = \vec{\beta}_q - S_qQ_q^T\vec{h} \\ &= \vec{\alpha}_{qj} - [\vec{S}_{1q},S_{2q},\ldots,\vec{S}_{jq}][\vec{Q}_{1q}^T\vec{h},\vec{Q}_{2q}^T\vec{h},\ldots,\vec{Q}_{jq}^T\vec{h}]^T\end{aligned}$$

Making the appropriate substitutions in terms of vectors from Grey's method and letting $r = (q-1)j$

$$\begin{aligned}\vec{\beta}_{q+1} &= \vec{\alpha}_{qj} - [\vec{C}_{r+1},\vec{C}_{r+2},\ldots,\vec{C}_{r+j}][\Delta\gamma_{r+1},\Delta\gamma_{r+2},\ldots,\Delta\gamma_{r+j}]^T \\ &= \vec{\alpha}_{qj} - \Delta\gamma_{r+1}\vec{C}_{r+1} - \Delta\gamma_{r+2}\vec{C}_{r+2} - \ldots - \Delta\gamma_{r+j}\vec{C}_{r+j} \\ &= \vec{\alpha}_{(q+1)j} .\end{aligned}$$

Thus, the induction is complete and the theorem is proven.

Q.E.D.

For the linear least squares problem Theorem 3.2 guarantees that, at least theoretically, one pass through the matrix of partials using the blocked algorithm (m/j steps) produces the same result as one pass through the matrix of partials using Grey's method (m steps).

4. Convergence Properties

By studying the local convergence properties of the blocked algorithm applied to the linear problem, one can get an idea about the convergence of the algorithm when it is being applied to the nonlinear regression problem. Broste has shown that for the linear problem, Grey's method will converge in at most m steps, i.e. property Q. From Theorem 3.2 it is clear that the point produced by m steps of Grey's method is identical to that produced by the blocked algorithm after m/j steps. Hence, the blocked orthogonalization algorithm also possesses property Q.

As Theorem 3.2 predicts, the change in the objective function when moving from a given point to the next will in general be greater when applying the blocked algorithm than when applying Grey's method. Theorem 3.3 tells what the change will be for each method as well as giving a way of comparing the function changes produced by the methods.

<u>THEOREM 3.3</u> Let $J(\vec{\alpha})$ be a linear least squares objective function. Then for a given point $\vec{\alpha}_{i-1}$

$$J(\vec{\alpha}_i) = (1 - \frac{||\Delta\vec{\gamma}_i||^2}{J(\vec{\alpha}_{i-1})})\ J(\vec{\alpha}_{i-1})$$

where $\Delta\vec{\gamma}_i$ is the change produced in the $\Delta\vec{\gamma}$-space by the i-th step of the blocked orthogonalization process.

PROOF Using the results as well as the proof of Lemma 3.2, it is clear that for the linear least squares objective

$$\frac{J(\vec{\alpha}_{i-1}) - J(\vec{\alpha}_i)}{J(\vec{\alpha}_{i-1})} = \frac{||\Delta\vec{\gamma}_i||^2}{J(\vec{\alpha}_{i-1})} .$$

Hence

$$J(\vec{\alpha}_i) = (1 - \frac{||\Delta\vec{\gamma}_i||^2}{J(\vec{\alpha}_{i-1})})\ J(\vec{\alpha}_{i-1})$$

where $\Delta\vec{\gamma}_i = -G_i^T\vec{h}(\vec{\alpha}_{i-1})$ is the j element vector denoting the change produced in the $\Delta\vec{\gamma}$-space by the i-th step of the blocked orthogonalization process in which there are j parameters per block.

Q.E.D.

Letting $\Delta\gamma_{ki}$ be the k-th element in the j element vector $\Delta\vec{\gamma}_i$, and noting that

$$||\Delta\vec{\gamma}_i|| = \sum_{k=1}^{j} \Delta\gamma_{ki}^2$$

one can see that at any given point $\vec{\alpha}_{i-1}$, the larger the value of $||\Delta\vec{\gamma}_i||^2$, the greater the reduction of the objective function. Further, the proof of Theorem 3.2 shows that the vector $\Delta\vec{\gamma}_i$ produced by the blocked algorithm has as its components, the changes produced in the orthogonal space by j applications of Grey's algorithm beginning at the point $\vec{\alpha}_{i-1}$. Thus, the reduction of the objective function when moving from the point $\vec{\alpha}_{i-1}$ to $\vec{\alpha}_i$ is a function of j, the number of parameters in each block. Theorem 3.3 then guarantees that the reduction produced by one application of the blocked method with $j > 1$ will at least be as good as that produced by Grey's method. Indeed, the reduction will be greater for the blocked method provided $\Delta\gamma_{ki} \neq 0$ for all $k = 2,3,\ldots,j$.

Although the following corollary is not surprising, it is of interest in its own right since it provides the value of the objective function in terms of the steps made in the orthogonal space. The proof is simply an application of Theorem 3.3 to the points $\vec{\alpha}_0, \vec{\alpha}_1, \ldots, \vec{\alpha}_{i-1}$.

COROLLARY 3.1 Let $J(\vec{\alpha})$ and $\Delta\vec{\gamma}_i$ be as in Theorem 3.3. Then

$$J(\vec{\alpha}_i) = J(\vec{\alpha}_0) - \sum_{k=1}^{i} ||\Delta\vec{\gamma}_k||^2$$

for $i = 1,2,\ldots,m/j$.

As the preceeding discussion implies, the greater the value of j, the greater the reduction in the objective function when moving from $\vec{\alpha}_{i-1}$ to $\vec{\alpha}_i$. It would seem desirable then that one should choose j = m, i.e. the Gauss-Hartley implementation, in order to produce the maximum stepwise reduction in the objective function. In this special case, one application of the blocked algorithm is equivalent to m applications of Grey's method. Hence, convergence occurs in one step.

While choosing j = m is definitely the optimal choice for j as far as the linear least squares problem is concerned, the computational results of the next chapter verify that for the nonlinear problem, such a choice may require more operations and function evaluations to obtain convergence than would be needed by choosing $j < m$.

Applying the blocked algorithm to nonlinear problems raises new questions which can no longer be answered by considering local convergence properties alone. Since the nonlinear problem causes the algorithm to become an infinite process, one is immediately faced with the problem of convergence. Broste [56] was able to show that Grey's algorithm would converge provided the initial parameter estimates were "sufficiently" close to the solution. His proof is also valid for the blocked algorithm.

In his Ph.D. dissertation, Cornwell [68] was able to prove that Grey's method would converge from any point in the parameter space provided a spacer step such as his Linear Acceleration Technique is included in the algorithm after every m steps.* When $j \neq m$, Cornwell's same proof can be used to guarantee convergence of the blocked orthogonalization method provided a spacer step is applied at every m/j steps of the process. Appendix F contains the mathematical framework used by Cornwell in proving convergence of his GOOP-LAT algorithm.

*In the absence of further restrictive assumptions, all convergence discussions refer to local minima.

IV. NUMERICAL INVESTIGATIONS

This chpater reports computational experience with the blocked orthogonalization algorithm when it was applied to problems whose very form indicate a natural grouping of the parameters being estimated. Since Cornwell's Linear Acceleration Technique [11,68] serves as both a theoretical safeguard and a practical acceleration trick, the blocked orthogonalization algorithm was studied with and without this modification.

The results reported here provide a comparison of the blocked orthogonalization algorithm with the well-known procedures of Grey, Gauss-Hartley, Levenberg-Marquardt, and Davidon-Fletcher-Powell. Various measures of effectiveness were used to measure algorithm performance. The results provide a uniform comparison of the algorithms in the sense that all algorithms were run on the same machine with code written by the same programmer. Recent studies such as that of Himmelblau [28] indicate the desirablilty for this type of consistency in comparing algorithm performance.

Attention is given first to a brief description of algorithm implementations. Following a discussion of the criteria selected for evaluating the algorithms, a careful description of each problem is given along with the results obtained from the numerical investigations.

A. Algorithms Implemented

Since various algorithms approach the nonlinear regression problem from different points of view, the blocked algorithm (BG) was tested against several different types of nonlinear programming methods. To this end, the Davidon-Fletcher-Powell (DFP), Levenberg-Marquardt (LM), Grey (GOOP), and Gauss-Hartley (GH) algorithms were chosen. These well-known algorithms were described in Chapters II and III. It should be noted that all algorithms except DFP are from the class of Least Squares Methods.

The blocked orthogonalization algorithm and Grey's method were tested with and without the acceleration technique suggested by Cornwell [11,68]. To ascertain the degree of accuracy required in the one-dimensional search routine used in this acceleration step, the step was implemented using Cornwell's original LAT and the modified quadratic fit of Aoki [69] which was already in use in the BG and GOOP routines. The BG-LAT and GOOP-LAT routines use LAT to implement the acceleration step in BG and GOOP respectively while BG-QF and GOOP-QF implement the acceleration step using the modified quadratic fit. Appendices A and C present the flow charts for the quadratic fit and the LAT algorithms respectively.

In an attempt to provide uniformity of test results and reduce the effects of programmer influence, careful efforts were made to duplicate as much code as possible between the algorithms. Indeed, the codes of the blocked algorithms, Grey's algorithms, and the Gauss-Hartley algorithm are identical. Appendix G contains a description of the details peculiar to the implementation of each of the methods tested.

All algorithms were run in double precision in Fortran H on the IBM 370/165. This provided the equivalent of sixteen significant digits of accuracy and helped to eliminate round-off and truncation problems which plague matrix operations and the Gram-Schmidt orthonormalization process. In the event one prefers to program the algorithms in single precision, the appropriate matrix operation precautions are advised. Since the classic Gram-Schmidt process tends to behave poorly on a matrix that is ill-conditioned, it is advisable to use a correction procedure as suggested by Mitchell and McCraith [70] when programming this orthogonalization process in single precision or to use extended precision for inner products.

Analytic derivatives were used in programming all algorithms reported. This is contrary to Grey's original implementation of GOOP to problems in lens design. Since such functions must be evaluated by ray tracing, the analytic calculation of derivatives is not feasible. For this reason, Grey used differences to estimate the partial derivatives. The reader is referred to Broste [56] for a discussion of implementing GOOP with finite differences. For the purposes of this investigation, it was felt that the use of analytic derivatives in all algorithms provided a more uniform comparison.

In applying algorithms to the test problems, each iterative process was allowed to continue until one of the following occurred:

1.) $||\vec{h}(\vec{\alpha}_i)|| \leq 10^{-10}$,

2.) No further reduction in the value of the objective function was possible. (At this point, the algorithm in question had found a local minimum.), or

3.) The number of steps taken had reached 400.

Each algorithm implementation checked the criteria in the order listed.

B. Comparison Criteria

The selection of criteria for algorithm comparison was complicated by the fact that each algorithm approaches the nonlinear problem from a different point of view. In addition, features of an algorithm which may be of importance to one user may be unimportant to another.

Indeed, desirable algorithm properties may vary from problem to problem. For example, if the objective function and its partials are difficult to evaluate, a user might desire an algorithm which keeps their evaluation to a minimum although other criteria such as the number of arithmetic operations necessary for convergence may be increased. For these reasons, several criteria were chosen for use in comparing algorithm performances.

1. Step Count

A step is said to have been made whenever part or all of the vector of parameters has been updated. In this context, the DFP, LM, and GH algorithms are total step methods since any update of parameters involves all parameters in the model. The various BG and GOOP algorithms tested are called partial step methods since only a part of the model parameters are updated at each step.

Since any updating of parameter estimates requires at least one call to the function and partial derivative routines, step counts are related to function evaluations and partial derivative counts. In the total step methods, each step requires a complete re-evaluation of the matrix of partial derivatives. This is not true for the partial step methods.

2. Function and Partial Derivative Evaluations

In cases where the objective function or the partial derivatives are difficult or require extensive operations to evaluate, a comparison using function and partial derivative evaluations may be of prime interest.

In the recorded data, each function evaluation (F.E.) represents one call to the function subprogram while each partial derivative evaluation (P.D.E.) represents the computation of one element in the nxm matrix of partial derivatives. These numbers represent actual program counts.

3. Operations Counts

Two types of operations counts were conducted for each problem. External operations counts (E.O.C.) represent the number of operations done outside the evaluation of the function and its partial derivatives while those done in evaluating the function and partial derivatives are reported as internal operations counts (I.O.C.). E.O.C. are representative of the complexity of the algorithm while I.O.C. represent function and partial derivative complexity.

While these counts are influenced by the individual programmer, such counts are helpful in comparing algorithms implemented by the same programmer. In the algorithm implementations used here, care was taken to avoid needless operations such as can occur in the calculation of $C_i \Delta \vec{\gamma}_i$ in the blocked algorithm. Since only the first ij rows of C_i are nonzero, the multiplication was done only through these rows. Further, quantities calculated in the function evaluation and used again in partial derivative evaluations were counted only once.

Since accurate time measurements were unavailable at the installation where these problems were run, total operations comparisons provide a rough comparison of the times required by various algorithms.

4. Local Minima

In practice, the model parameters being estimated may be subject to constraints which are implied but not explicitly stated in the model formulation. For example, some parameters in the YAG problem are estimates of peak amplitude. Since such parameters must be nonnegative, any local minima containing negative values for the parameters corresponding to peak amplitude were not acceptable. Hence, before deciding to accept or reject a local minimum, both the function value and the parameter values were checked for feasibility.

C. Test Problems and Results

The numerical results reported in this section were obtained from models in which the parameters being estimated were naturally grouped. Some of the problems cited have their origins in the "real world" while others were contrived from linear combinations of well-known functions.

Each problem was tested using the algorithm implementations in Appendix G. In all problems except one, tests were performed using different starting values for the same problem.

1. Generated YAG Problems

The problems reported in this section are those of the YAG type of Chapter I, equation (1.5). They were generated in the sense that parameter values and hence, peak locations, shapes, and sizes were chosen first. Then, corresponding sets of fifty-one data points were generated for each problem. The examples represented cases in which the data displayed two, three and four peaks respectively. For all cases, the peaks were chosen in close proximity to each other as experience has shown this type of problem to be more difficult than problems where peaks are widely separated. (The blocked aberration vectors are more nearly orthogonal when peaks are widely separated.) Recalling the relationship between parameters and the physical appearance of the data peaks, as was mentioned in Chapter III, parameters for the BG algorithm were grouped into blocks of three, each block corresponding to a peak.

Table I contains the solutions and initial starting points used for problems 1-6 while Table II lists the results of each algorithm. In problems 4 and 5, non-negativity restrictions were placed on the parameters associated with peak amplitude (i.e. the first, fourth, seventh, and tenth parameters) in all algorithms except DFP and LM. (One feature of the present algorithm is the ease by which variables may be constrained.) A similar approach was used by Lesnick and Rigler [9] in their work with this type of problem.

Table III contains a list of solutions produced by algorithms in which execution was terminated at 400 steps or when a point was reached for which the algorithm failed to further reduce the value of the objective function.

An initial evaluation of the results of these tests reveal that when convergence occurs, the LM, GH, and BG-LAT algorithms converge at an impressive rate. However, recall that LM and GH are total step methods. Hence, each step taken requires complete re-evaluation of the matrix of partial derivatives. BG-LAT requires re-evaluation of only three columns of the partial derivative matrix for every step taken.

Table I

Initial and Solution Values for Generated YAG Problems

Solution	Problem #1	Problem #2	Solution	Problem #3	Problem #4	Solution	Problem #5	Problem #6
0.50	0.45	0.45	0.50	0.45	0.45	0.50	0.45	0.60
0.50	0.55	0.55	0.50	0.55	0.55	0.50	0.55	0.40
2.50	2.50	2.00	2.50	2.00	2.00	2.50	2.00	2.60
4.00	4.50	4.50	4.00	4.50	1.125	4.00	4.50	3.50
1.00	0.80	0.80	1.00	0.80	0.40	1.00	0.08	1.20
3.50	3.50	4.00	3.50	4.00	4.00	3.50	4.50	3.70
			0.25	0.30	0.30	0.25	0.20	0.40
			0.50	0.40	0.40	0.50	0.60	0.40
			4.50	5.00	5.00	4.50	4.10	5.00
						3.00	3.10	2.50
						1.00	0.90	1.50
						6.00	6.20	5.50

Table II

Experimental Results for Generated YAG Problems

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
1	DFP	30	75	22950	11450	130050	$6\text{x}10^{-12}$
	LM	4	5	1220	17960	5200	$6\text{x}10^{-12}$
	GH	6	17	1840	40490	14620	$6\text{x}10^{-11}$
	GOOP	107	278	5460	106580	209360	$7\text{x}10^{-11}$
	GOOP-LAT	72	226	3670	72780	168710	$5\text{x}10^{-11}$
	GOOP-QF	57	184	2910	56630	137190	$8\text{x}10^{-11}$
	BG	30	74	4590	89150	59100	$7\text{x}10^{-11}$
	BG-LAT	22	83	3370	65810	63920	$6\text{x}10^{-12}$
	BG-QF	22	86	3370	66090	66060	$1\text{x}10^{-10}$

Table II (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
2	DFP	30	74	22640	11430	128320	$2x10^{-11}$
	LM	10	18	3060	46280	16900	$8x10^{-12}$
	GH	20	44	6120	134830	39580	$5x10^{-11}$
	GOOP	79	208	4030	78530	156450	$6x10^{-12}$
	GOOP-LAT	55	165	2810	54780	121810	$3x10^{-11}$
	GOOP-QF	81	249	4130	80880	186050	$6x10^{-11}$
	BG	41	94	6270	120740	75670	$1x10^{-10}$
	BG-LAT	21	79	3210	61740	60850	$1x10^{-11}$
	BG-QF	23	90	3520	68070	69130	$6x10^{-11}$

Table II (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
3	DFP	54	142	65180	42280	362100	5×10^{-11}
	LM	48	Local minimum				0.28
	GH	5	Local minimum				2.29
	GOOP	150	439	7650	209360	459300	6×10^{-11}
	GOOP-LAT	117	370	5970	165830	383140	6×10^{-12}
	GOOP-QF	126	406	6430	179110	423910	6×10^{-12}
	BG	39	113	5970	157800	123560	6×10^{-12}
	BG-LAT	30	110	4590	121910	118650	6×10^{-12}
	BG-QF	36	131	5510	146600	141360	6×10^{-12}

Table II (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
4	DFP	400	Execution terminated				0.18
	LM	32	61	14690	320000	81770	$6\text{x}10^{-11}$
	GH	19	55	8720	277620	67890	$6\text{x}10^{-12}$
	GOOP	213	628	10860	298380	662290	$6\text{x}10^{-11}$
	GOOP-LAT	144	459	7340	204170	482870	$6\text{x}10^{-12}$
	GOOP-QF	153	496	7800	217550	521530	$7\text{x}10^{-11}$
	BG	54	160	8260	218630	174700	$6\text{x}10^{-12}$
	BG-LAT	36	134	5510	146330	144430	$6\text{x}10^{-12}$
	BG-QF	36	140	5510	146770	150560	$6\text{x}10^{-12}$

Table II (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
5	DFP	108	301	184210	143960	1013170	$1x10^{-11}$
	LM	5	Local minimum				.4
	GH	0	Local minimum				135.85
	GOOP	400	Execution terminated				.001
	GOOP-LAT	292	1063	14890	533160	1439220	$5x10^{-11}$
	GOOP-QF	324	1119	16520	595940	1516840	$7x10^{-11}$
	BG	264	856	40390	1357050	1215840	$3x10^{-11}$
	BG-LAT	128	516	19580	600850	723384	$2x10^{-11}$
	BG-QF	198	Local minimum				5.2

Table II (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
6	DFP	144	398	243580	191820	1339670	5×10^{-11}
	LM	20	35	12240	344210	62710	1×10^{-11}
	GH	30	100	18360	768860	764980	1×10^{-11}
	GOOP	400	Execution terminated				0.21
	GOOP-LAT	400	Execution terminated				0.20
	GOOP-QF	400	Execution terminated				0.20
	BG	148	448	22640	759930	626032	1×10^{-11}
	BG-LAT	84	315	12852	432990	436090	2×10^{-11}
	BG-QF	100	372	15300	516250	515160	1×10^{-11}

Table III

Local Minima Produced in Solving Generated YAG Problems

	Problem #3		Problem #4
Solution	LM	GH	DFP
0.50	0.53	0.93	-4.12
0.50	0.51	0.52	1.03
2.50	2.56	2.61	2.36
4.00	5.78	3.83	5.50
1.00	1.17	1.02	0.92
3.50	3.70	3.75	2.51
0.25	-0.53	0.03	3.72
0.50	1.50	1.39	1.03
4.50	5.96	4.25	3.82

Table III (Continued)

	Problem #5			Problem #6		
Solution	LM	GOOP	BG-QF	GOOP	GOOP-LAT	GOOP-QF
0.50	1.13	0.47	2.19	0.52	0.51	0.51
0.50	0.69	0.49	0.79	0.50	0.50	0.50
2.50	2.67	2.50	2.76	2.54	2.54	2.54
4.00	5.88	0.22	1.79	4.84	4.60	4.60
1.00	1.23	0.48	0.78	1.10	1.08	1.08
3.50	3.85	4.50	3.98	3.63	3.61	3.61
0.25	$-4\text{x}10^{5}$	4.17	0.23	-1.07	-3.54	-2.75
0.50	$-2\text{x}10^{7}$	1.02	0.77	0.65	0.74	0.72
4.50	$6\text{x}10^{8}$	3.50	4.03	5.57	5.57	5.57
3.00	2.10	3.00	3.04	4.64	7.41	6.58
1.00	0.88	1.00	0.90	0.95	0.90	0.91
6.00	6.06	6.00	6.20	5.74	5.65	5.67

While it was predicted *a priori* that GOOP-QF and BG-QF would require more function evaluations than GOOP-LAT and BG-LAT, it was expected that the former routines would require fewer steps for convergence since the acceleration step used a more accurate search. The experimental results indicate that LAT proved to be a better algorithm for acceleration of convergence than QF.

In examination of the local minima stated in Table III, recall that the physical significance of all model parameters require that they be positive. Hence, solutions with negative parameters are totally unacceptable. Recall that parameters were restricted to be nonnegative in problems four and five for all but the DFP and LM algorithms.

Before reporting the LM results for problem five, numerous large and small damping factors were tested in an attempt to obtain convergence. In each case, the algorithm terminated at a local minimum in which parameter values were negative. The results reported used a damping factor of 0.01 and represent the local minimum having the smallest function value.

The GOOP results for problem five are interesting in that the parameters corresponding to peak locations two and three have been switched. Had execution not been terminated at 400 steps, it is expected that convergence would have occurred as exhibited by the parameter and function values. Note that in selection of the initial starting values used in problem five, the order of the second and third peaks was switched. Indeed, the location of the second peak was chosen as 4.50 which is the actual location of the third peak in the data. This peak switching did not occur in the local minimum produced by BG-QF.

Note that in problem six, the blocked algorithms were able to maintain parameter relationships and to converge while the GOOP versions produced minima with negative parameters.

2. Spectroscopic Analysis Problem

The spectroscopic analysis problem used thirty-seven points of actual YAG data obtained from a spectroscopic analysis experiment. The data with its six peaks is plotted in Figure 1. Not only did this data contain considerable noise, but the problem was further complicated by the presence of instrument gain as the data was plotted from left to right along the abscissa. For this reason, a quadratic term was added to the model of equation (1.5) resulting in the model

$$y = \sum_{j=1}^{6} \frac{\alpha_{3j-2}}{\alpha_{3j-1}^2 + (\alpha_{3j} - x)^2} + \alpha_{19}x^2 + \alpha_{20}x + \alpha_{21} + \varepsilon .$$

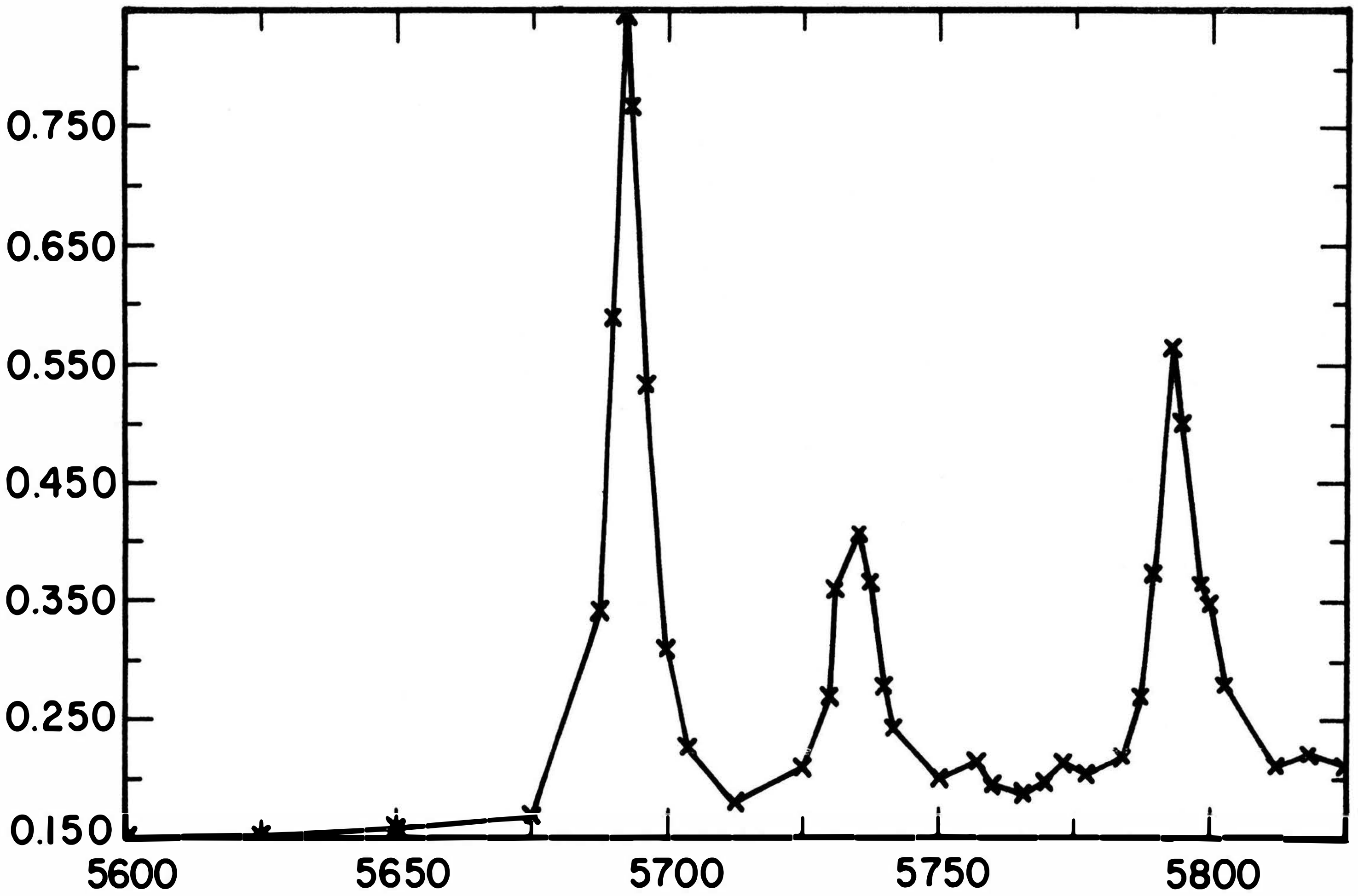


Figure 1.--Graph of Spectroscopic Analysis Data

The three additional parameters created by the introduction of the quadratic term were taken to comprise an additional block of parameters when the problem was solved by the blocked algorithms.

In selecting the initial parameter values reported in Table IV, careful consideration was given to the relationships between parameter values and peak amplitude, half-width, and location. All algorithms except DFP and LM placed nonnegativity constraints on all parameters except those associated with the quadratic shift factor.

This problem differed from the others tested since the solution was unknown and the data being fitted contained noise. Table V records results of the comparison criteria. As Table V indicates, DFP produced the smallest function value, but required 190480 partial derivative evaluations compared with 4330 for BG. Only GOOP-QF exceeded DFP in the total number of operations required. Note that LM and GH, the other two total step methods, were completely ineffective. Various damping factors were tried in the LM algorithm, but all solutions contained negative parameters.

All three versions of GOOP appear to be successful until one examines the GOOP-LAT solution in Table IV. Table IV indicates that peaks three and four have been switched, a highly questionable result in view of Figure 1.

Table IV

Initial and Computed Solutions for Spectroscopic Analysis Problem

Initial Values	DFP	LM	GH
6.80	7.94	7.51	6.80
3.30	3.44	3.36	3.30
5693.00	5692.69	5692.60	5693.00
3.50	4.63	4.46	3.50
4.20	4.31	4.29	4.20
5735.00	5734.70	5734.60	5735.00
1.20	0.47	-1.24	1.20
5.20	5.48	-72.50	5.200
5756.00	5755.92	5753.70	5756.00
0.20	0.09	-0.21	0.20
3.20	3.34	3.96	3.20
5773.00	5772.98	5774.60	5773.00
3.48	5.83	5.08	3.48
3.30	4.01	3.83	3.30
5794.00	5793.92	5793.90	5794.00
2.5×10^{-5}	9×10^{-3}	-3101.7	2.5×10^{-5}
0.05	0.39	1.5×10^{16}	.05
5800.00	5800.12	3.8×10^{10}	5800.00
0.0	-1.3×10^{-8}	-1×10^{-8}	0.0
3×10^{-4}	3.5×10^{-4}	2.99×10^{-4}	3×10^{-4}
-1.53	-1.44	-1.53	-1.53

Table IV (Continued)

GOOP	GOOP-LAT	GOOP-QF
7.21	7.21	7.22
3.31	3.31	3.31
5692.64	5692.64	5692.64
3.74	3.75	3.77
3.97	3.97	3.99
5734.54	5734.54	5734.54
$2x10^{-4}$	0.18	$4x10^{-3}$
6.11	80.86	0.10
5755.57	5782.95	5756.19
0.02	0.04	$4x10^{-5}$
1.09	1.31	3.27
5771.91	5772.21	5774.80
3.59	3.42	3.22
3.24	3.16	3.15
5793.79	5793.76	5793.75
$1x10^{-4}$	$1x10^{-4}$	$1x10^{-4}$
0.05	0.05	0.05
5800.00	5800.00	5800.00
0.0	$4x10^{-14}$	$8x10^{-17}$
$3x10^{-4}$	$3x10^{-4}$	$3x10^{-4}$
-1.53	-1.53	-1.53

Table IV (Continued)

BG	BG-LAT	BG-QF
7.21	7.21	7.21
3.31	3.31	3.31
5692.64	5692.64	5692.64
3.74	3.74	3.74
3.97	3.97	3.97
5734.54	5734.54	5734.54
$2x10^{-3}$	$2x10^{-3}$	$4x10^{-3}$
1.10	1.14	1.21
5756.67	5756.72	5756.60
0.01	$9x10^{-3}$	$1x10^{-2}$
0.07	0.05	0.17
5771.98	5771.96	5771.97
4.02	4.02	4.02
3.52	3.52	3.52
5793.96	5793.96	5793.96
$2.5x10^{-5}$	$2.5x10^{-5}$	$2.5x10^{-5}$
.05	.05	.05
5800.00	5800.00	5800.00
0.0	0.0	0.0
$3x10^{-4}$	$3x10^{-4}$	$3x10^{-4}$
-1.53	-1.53	-1.53

Table V

Experimental Results for Spectroscopic Analysis Problem

Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
DFP	48	286	190480	194140	1153150	0.010
LM	2	Execution terminated due to overflow				0.014
GH	29	132	3220	180930	215560	0.0354
GOOP	94	355	3480	206900	570800	0.0149
GOOP-LAT	106	432	3920	245320	693970	0.0154
GOOP-QF	239	979	8840	547506	1572800	0.0171
BG	39	165	4330	238900	270063	0.0169
BG-LAT	29	130	3220	180610	212380	0.0169
BG-QF	29	132	3220	180930	215560	0.0169

While the results of the blocked algorithms were not quite as good as GOOP, Table V indicates rapid convergence. The accelerated versions performed about the same. The three blocked algorithms produced the same function value but different solutions.

3. Airplane Problem

The mathematical model under consideration in this problem comes from the study of damped vibrations and was suggested by a similar model used in the design and testing of aircraft. The test function takes the form

$$y = \sum_{j=1}^{r} \alpha_{4j-3} e^{-\alpha_{4j-2} t} \cos(\alpha_{4j-1} t + \alpha_{4j}) + \varepsilon$$

where r represents both the number of terms in the function as well as the number of peaks in the test data. In applying the blocked algorithms to this problem, four parameters were included in each block. No bounds were placed on any of the parameters. Fifty data points were generated using the solution values shown in Table VI and allowing the independent time parameter t to run from zero to five seconds.

Examination of the test results in Table VII indicate the difficulty of this problem particularly for the DFP and GOOP algorithms. Even LAT was ineffective in accelerating

Table VI

Initial and Solution Values for Airplane Problem

Solution Values	Problem #1	Problem #2
3.00	3.04	2.90
3.00	2.95	3.15
2.25	2.23	2.10
-1.00	-0.95	-1.05
4.50	4.55	4.45
1.50	1.45	1.55
1.75	1.80	1.65
-0.50	-0.45	-0.55

GOOP in problem #1. The performances of LM, GH, and the blocked algorithms represent substantial savings in computer time in view of the exponential, sin, and cos evaluations necessary in evaluating the function and its partial derivatives.

4. Normal Problem

The normal problem represents an attempt to fit data using a linear combination of normal functions, viz.

Table VII

Experimental Results for Airplane Problem

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
1	DFP	142	1052	420800	114270	1998800	$1x10^{-10}$
	LM	4	5	1600	30210	4700	$6x10^{-12}$
	GH	4	13	1600	45600	10300	$2x10^{-19}$
	GOOP	169	500	8450	210800	356350	$1x10^{-11}$
	GOOP-LAT	169	560	8450	211940	398350	$7x10^{-12}$
	GOOP-QF	136	467	6800	172020	332000	$2x10^{-11}$
	BG	25	79	5000	122330	59050	$3x10^{-13}$
	BG-LAT	13	57	2600	63040	41850	$6x10^{-12}$
	BG-QF	18	80	3600	90070	58700	$1x10^{-14}$

Table VII (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
2	DFP	104	640	256000	78490	1157120	$1x10^{-10}$
	LM	4	5	1600	30210	4700	$4x10^{-11}$
	GH	4	13	1600	43900	10300	$5x10^{-14}$
	GOOP	272	835	13600	340610	594700	$5x10^{-11}$
	GOOP-LAT	184	639	9200	231690	454200	$9x10^{-11}$
	GOOP-QF	249	843	12450	313150	599450	$7x10^{-11}$
	BG	25	76	5000	119910	56950	$5x10^{-13}$
	BG-LAT	15	66	3000	71490	48450	$2x10^{-11}$
	BG-QF	14	57	2800	68610	42000	$7x10^{-11}$

$$y = \sum_{j=1}^{r} \alpha_{3j-2} e^{-0.5 q_j^2(x,\vec{\alpha})} + \varepsilon$$

where

$$q_j(x,\vec{\alpha}) = \frac{x - \alpha_{3j-1}}{\alpha_{3j}}$$

and r corresponds to the number of terms in the function as well as the number of peaks in the data. The following estimates are available for the j-th data peak

amplitude:	α_{3j-2}
half-width:	$\alpha_{3j}\sqrt{2\ln 2}$
peak location:	α_{3j-1}.

Forty data points were generated over the interval -2.5 to 3.0 using the solution values shown in Table VIII.

Algorithms were applied using the initial values shown in Table VIII. In application of the blocked algorithms, the model parameters were separated into groups of three. No bounds were placed on any of the parameters.

The results recorded in Table IX indicate LM, GH, and BG-LAT are the most effective algorithms in solving this problem. In both problems, LAT produced faster

Table VIII

Initial and Solution Values for Normal Problem

Solution Values	Problem #1	Problem #2
0.75	1.00	0.40
-1.00	-0.75	-1.25
0.50	0.40	0.75
1.25	1.00	1.55
1.50	1.00	2.00
1.00	1.25	1.40

convergence than QF. All versions of the blocked algorithm outperformed the GOOP versions. In problem #1, the minimum function value was produced by LM. BG-LAT produced the minimum function value in problem #2.

5. Exponential Problem

The exponential problem often arises in the physical and biological sciences. The model is composed of a linear combination of exponential functions, viz.

$$y = \sum_{j=1}^{r} \alpha_{2j-1} e^{-\alpha_{2j} x} + \varepsilon .$$

Table IX

Experimental Results for Normal Problem

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
1	DFP	18	38	9120	6670	47350	$5x10^{-14}$
	LM	5	6	1200	17830	4160	$1x10^{-19}$
	GH	7	18	1680	36630	11760	$1x10^{-15}$
	GOOP	88	199	3520	68830	116080	$8x10^{-11}$
	GOOP-LAT	41	120	1640	32290	69360	$2x10^{-12}$
	GOOP-QF	58	182	2320	45880	104960	$6x10^{-11}$
	BG	27	60	3240	62150	36840	$8x10^{-11}$
	BG-LAT	11	40	1320	25020	23720	$2x10^{-11}$
	BG-QF	21	84	2520	48900	49560	$4x10^{-11}$

Table IX (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
2	DFP	18	37	8880	6650	46100	$1x10^{-13}$
	LM	5	6	1200	17830	4160	$6x10^{-14}$
	GH	18	37	4320	94400	25040	$5x10^{-11}$
	GOOP	93	250	3720	72960	144960	$6x10^{-11}$
	GOOP-LAT	43	127	1720	33950	73360	$2x10^{-12}$
	GOOP-QF	58	180	2320	45890	103840	$3x10^{-11}$
	BG	36	75	4320	84020	46320	$4x10^{-11}$
	BG-LAT	10	35	1200	23460	20800	$6x10^{-17}$
	BG-QF	12	46	1440	28440	27200	$4x10^{-12}$

Its inclusion here was suggested by Osborne [63] who solved this type of problem by separating the variables into groups of linear and nonlinear parameters. Forty data points were generated over the interval -1.0 to 5.0 using the solution values shown in Table X.

Table X

Initial and Solution Values
for Exponential Problem

Solution Values	Problem #1	Problem #2
0.50	0.30	0.60
3.20	2.50	3.10
1.00	1.20	0.90
0.50	0.60	0.40
0.70	0.50	1.00
-0.50	-0.70	-0.40

Algorithms were applied using the initial values shown in Table X. In application of the blocked algorithms, two parameters were placed in each block. No bounds were placed on any of the parameters.

The results recorded in Table XI indicate LM, GH, and BG-LAT are superior in solving this problem. In problem #1, QF provided better acceleration for GOOP

Table XI

Experimental Results for Exponential Problem

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
1	DFP	45	98	23520	16710	102508	$2\text{x}10^{-11}$
	LM	8	12	1920	29120	10560	$7\text{x}10^{-18}$
	GH	20	43	4800	107100	21320	$6\text{x}10^{-11}$
	GOOP	206	Local minimum				26.0
	GOOP-LAT	103	331	4120	82740	147680	$7\text{x}10^{-11}$
	GOOP-QF	91	314	3640	73650	139960	$7\text{x}10^{-11}$
	BG	94	Local minimum				12.3
	BG-LAT	31	106	2480	47260	47880	$1\text{x}10^{-12}$
	BG-QF	42	148	3360	65460	66800	$4\text{x}10^{-11}$

Table XI (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
2	DFP	27	59	14160	9930	61710	$8\text{x}10^{-11}$
	LM	4	5	960	14260	4480	$1\text{x}10^{-12}$
	GH	19	40	4560	101740	19880	$3\text{x}10^{-11}$
	GOOP	119	287	4760	94400	128640	$7\text{x}10^{-11}$
	GOOP-LAT	49	148	1960	38960	66080	$7\text{x}10^{-11}$
	GOOP-QF	67	213	2680	53780	95040	$7\text{x}10^{-11}$
	BG	57	116	4560	87320	53320	$4\text{x}10^{-11}$
	BG-LAT	19	63	1520	28650	28480	$4\text{x}10^{-11}$
	BG-QF	25	88	2000	38200	39720	$3\text{x}10^{-11}$

than did LAT. This was reversed in problem #2. All accelerated versions of BG outperformed the GOOP versions. LM produced the minimum function value and required fewer partial derivative evaluations than did any of the other algorithms. The local minima produced in problem #1 by GOOP and BG are recorded in Table XII.

Table XII

Local Minima Produced in Solving Exponential Problem

Solution Values	Algorithm GOOP	BG
0.50	1.72	1.50
3.20	2.04	2.19
1.00	-0.075	-0.02
0.50	0.60	-0.87
0.70	0.37	0.47
-0.50	-0.69	-0.62

6. Trigonometric Series Problem

The model used in this problem is of the form

$$y = \sum_{j=1}^{r} \alpha_{3j-2} \cos \alpha_{3j-1} x + \alpha_{3j} \sin \alpha_{3j-1} x + \varepsilon$$

where r is again related to the number of terms in the model as well as the number of peaks in the data. Fifty data points were generated over the interval -1.5 to 9.5 using the starting values shown in Table XIII.

Table XIII

Initial and Solution Values for Trigonometric Series Problem

Solution Values	Problem #1	Problem #2
2.00	1.80	2.10
1.00	0.90	0.90
0.90	1.00	0.80
0.80	0.75	0.90
0.50	0.60	0.60
1.10	1.00	1.20

Algorithms were applied using the initial values shown in Table XIII. In application of the blocked algorithms, three parameters were placed in each block. No bounds were placed on any of the parameters.

The results recorded in Table XIV indicate LM, GH, BG-LAT, and BG-QF are superior in solving this problem. While BG-LAT requires more function evaluations than

Table XIV

Experimental Results for
Trigonometric Series Problem

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
1	DFP	21	44	13200	7630	61860	$8x10^{-11}$
	LM	5	6	1500	22030	4600	$2x10^{-13}$
	GH	6	19	1800	39760	12600	$3x10^{-16}$
	GOOP	109	281	5450	106430	172200	$7x10^{-11}$
	GOOP-LAT	50	153	2500	48520	93500	$4x10^{-14}$
	GOOP-QF	74	233	3700	72640	142300	$7x10^{-11}$
	BG	38	79	5700	110690	51200	$6x10^{-11}$
	BG-LAT	9	29	1350	25320	18300	$1x10^{-13}$
	BG-QF	10	38	1500	29450	23800	$5x10^{-16}$

Table XIV (Continued)

Problem	Method	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.	F.V.
2	DFP	24	52	15600	8950	73110	$6x10^{-12}$
	LM	5	6	1500	22030	4600	$1x10^{-16}$
	GH	6	18	1800	39740	12000	$2x10^{-19}$
	GOOP	97	232	4850	94590	142400	$7x10^{-12}$
	GOOP-LAT	49	141	2450	47790	86200	$3x10^{-19}$
	GOOP-QF	65	199	3250	63960	121600	$9x10^{-11}$
	BG	34	73	5100	99060	47200	$4x10^{-11}$
	BG-LAT	9	32	1350	25370	20100	$7x10^{-15}$
	BG-QF	10	38	1500	29450	23800	$4x10^{-17}$

either LM or GH, it required fewer partial derivative evaluations. All versions of the blocked algorithm outperformed the GOOP algorithms.

D. Summary of Numerical Investigations

Results of the numerical investigations were consolidated using a scheme similar to that suggested by Lessman [31]. For each problem tested, an algorithm was awarded nine points if it finished first, eight points if it finished second, etc., in each of the categories. Zero points were awarded for failure of an algorithm to converge to the prescribed minimum. Since answers to the Spectroscopic Analysis Problem were not known, it was not included in the computation of the results shown in Table XV. Table XVI presents a ranking of the algorithms for each of the comparison criteria. Table XVI was computed using the data in Table XV.

Examination of the results of Table XVI indicate the most favorable results for the number of steps, function evaluations, and internal operations counts were achieved by LM. Since all of these quantities are directly related, it is not surprising that the same algorithm was best in all three categories.

Table XV

Comparison of Algorithm Results

	Comparison Criteria				
	Steps	F.E.	D.P.E.	E.O.C.	I.O.C.
DFP	72	76	42	104	54
LM	107	107	99	84	105
GH	97	96	71	52	93
GOOP	23	18	40	21	19
GOOP-LAT	50	46	85	55	45
GOOP-QF	43	42	75	47	36
BG	67	62	58	46	73
BG-LAT	105	95	110	97	103
BG-QF	86	72	91	80	79

Since the matrix of partials is completely re-evaluated at each step, it is not surprising that a total step method was more successful in these three categories than were the partial step methods.

In all categories recorded, BG-LAT ranked in the upper third of the algorithms tested with BG-QF in the upper five ninths. BG-LAT's first place ranking in partial derivative evaluations suggests its usefulness on problems in which the partial derivatives require extensive calculations or are calculated using

Table XVI

Algorithm Rankings

Rank	Steps	F.E.	P.D.E.	E.O.C.	I.O.C.
1	LM	LM	BG-LAT	DFP	LM
2	BG-LAT	GH	LM	BG-LAT	BG-LAT
3	GH	BG-LAT	BG-QF	LM	GH
4	BG-QF	DFP	GOOP-LAT	BG-QF	BG-QF
5	DFP	BG-QF	GOOP-QF	GOOP-LAT	BG
6	BG	BG	GH	GH	DFP
7	GOOP-LAT	GOOP-LAT	BG	GOOP-QF	GOOP-LAT
8	GOOP-QF	GOOP-QF	DFP	BG	GOOP-QF
9	GOOP	GOOP	GOOP	GOOP	GOOP

finite differences. Since operations counts are related to execution times, BG-LAT's first place ranking in total operations counts as seen in Table XV indicate that it may require less execution time on many problems than would be required by the other algorithms.

As was predicted by Theorem 3.2, the performances of the blocked orthogonalization algorithm were better than those of GOOP. While the accelerated versions of BG and GOOP performed better than the unaccelerated versions, it was surprising to note that in most cases, the LAT acceleration step produced better results than

QF. It was conjectured that the reason for this might be that given a direction vector $\Delta\vec{\alpha}$ and a point $\vec{\alpha}$, the LAT algorithm finds a point at $\vec{\alpha} + t\Delta\vec{\alpha}$ where $t \geq 0$. The QF algorithm finds a point $\vec{\alpha} + t\Delta\vec{\alpha}$ but will allow $t < 0$ or $t \geq 0$. It appears that while QF may find a point of lower function value by allowing $t < 0$, this reversal of direction may serve to hinder convergence. In examination of all test results where LAT outperformed QF, there were several steps in QF in which $t < 0$.

While Tables XV and XVI rank the successful algorithm performances on fourteen of the fifteen problems tested, they give no direct indication of algorithm failures. Table XVII records the number of times each algorithm was terminated due to reaching the maximum number of steps or to finding a local minimum. The failure of LM in the Spectroscopic Analysis Problem has been included in this table.

As the results in Table XVII indicate, GOOP and LM had the largest number of forced terminations while BG-LAT had none.* In comparing the local minima in Table III, that produced by BG-QF appears to be more representative of the desired solution than any of the others listed.

*Several damping factors were tried in LM in attempts to reduce its failures.

Table XVII

Number of Forced Algorithm Terminations

Algorithm	# Terminations
DFP	1
LM	3
GH	2
GOOP	3
GOOP-LAT	2
GOOP-QF	1
BG	1
BG-LAT	0
BG-QF	1

The Spectroscopic Analysis Problem is of particular interest since it was the only problem tested in which the solution was unknown and in which the original data contained noise. In this case, GOOP slightly outperformed the blocked algorithms in reducing function value. However, GOOP-LAT converged to a solution in which two of the data peaks were switched! All three versions of the blocked algorithm produced the same function value.

The numerical investigations described in this chapter indicate the competitiveness of the blocked orthogonalization algorithms in solving nonlinear regression problems in which the parameters are naturally grouped. Indeed, BG-LAT showed a savings over the other algorithms in both partial derivative evaluations and total operations. In problems where the parameters were related to the physical appearances of the data peaks, blocking parameters appeared to preserve parameter relationships even when local minima were produced.

V. CONCLUSIONS

Grey's method [3,4] has been successfully used in many applied fields. Further, it is tedious but easy to verify that von Holdt's block orthogonalization [10] is effective in solving least squares problems. In attempts to find characteristics of nonlinear regression problems which can be identified a priori, an algorithm was designed which combined Grey's method with von Holdt's block orthogonalization. The resulting algorithm was shown to be effective in solving nonlinear regression problems in which the parameters to be estimated were naturally grouped. The new algorithm differs from current separation of variables algorithms in that each block of parameters may contain linear as well as nonlinear parameters. Hence, while the new algorithm is distinctly different from the separation of variables algorithms, it can be used to solve this type of problem.

From the theoretical and experimental results presented, it is clear that the blocked algorithm combines the best features of the GOOP and GH algorithms while avoiding their worst features. For cases in which the linearization produced by the Taylor series expansion is an exact representation of the function in question, it was shown that the set of points generated by BG was actually a subset of the set generated by GOOP. Since the present implementation of GH was accomplished by BG with j = m, the set of points produced by GH when applied to the linear problem is a subset of those produced by BG. It was then shown that for the linear problem, the relative rates of convergence are in decreasing order: GH, BG, and GOOP. The theoretical similarity of the blocked algorithm to GOOP gave it several desirable properties of that algorithm. As in GOOP, the blocked algorithm is capable of detecting and avoiding redundant parameters. Further, convergence of the new algorithm is guaranteed for the nonlinear problem as long as Cornwell's acceleration step [11,68] is inserted each time the matrix of partial derivatives has been updated. As with GOOP, BG is a conjugate direction algorithm.

While theoretical predictions were consistent with the numerical results, several important features appeared in the numerical tests which the theory did not predict. The first and perhaps most important feature was the blocked algorithm's robustness. Tests showed the blocked algorithm more consistent in converging to the global minima than was GOOP. Secondly, when local minima were produced by the algorithms, those produced by the blocked algorithms more nearly maintained the physical relationships between parameter values and data appearance. Hence, blocking parameters according to their natural grouping appeared to produce a more stable and dependable algorithm. Finally, it should be noted that the acceleration of GOOP produced by the various blocked algorithms did not come at the expense of additional operations or function evaluations. As indicated in Tables XV and XVI, the blocked versions outperformed the GOOP versions in all aspects of the tests except one. In all tests, BG-LAT was in the upper third of all algorithms tested.

From the numerical investigations cited, it appears that the blocked algorithm is very effective in solving linear and nonlinear regression problems in which the parameters are naturally grouped. It is particularly attractive for use with problems in which one desires to minimize the number of partial derivative evaluations. The algorithm's stability and dependability make it applicable to a wide range of problems.

Development of the blocked algorithm opens up several interesting areas for further research. For example, it would seem desirable to incorporate a natural blocking scheme into other algorithms such as DFP and LM. This idea is particularly appealing for conjugate direction algorithms such as the DFP. Further, since individual blocks of parameters often contain linear and nonlinear parameters, it might be desirable to apply a separation of variables technique to individual blocks of parameters.

In addition to testing the current blocked algorithm on more problems containing naturally grouped parameters, several variations of the current algorithm warrant further study. The algorithm should be modified and tested on problems in which the block sizes vary within the problem. This would require simple programming modifications only. A further modification calls for using LAT exclusively for all searches required in the accelerated blocked algorithm or to replace the one-dimensional searches by the two stage approximation Broste [56] used in GOOP. The convergence rate might be improved by permuting the order in which the blocks of parameters are evaluated at each iteration in the same way Lesnick and Rigler [9] did in their version of GOOP. Cornwell claimed that running GOOP in single precision produces results as acceptable as those obtained in double precision. This claim should be checked out with the new algorithm. Since GOOP was originally programmed using finite difference approximations for the partials, a similar approach needs to be tried with the blocked algorithm.

Further theoretical work with the algorithm is needed to determine its convergence rate for nonlinear problems as well as determining those properties of the natural grouping strategy which lead to the algorithm's robustness.

BIBLIOGRAPHY

1. Levenberg, K. "A Method for the Solution of Certain Non-Linear Problems in Least Squares," Quarterly of Applied Mathematics, Vol. 8, 1944, pp. 212-221.

2. Jamieson, T. H. Optimization Techniques in Lens Design. New York: American Elsevier, 1971.

3. Grey, D. S. "Aberration Theories for Semi-Automatic Lens Design by Electronic Computers. I. Preliminary Remarks," Journal of the Optical Society of America, Vol. 53, 1963, pp. 672-676.

4. Grey, D. S. "Aberration Theories for Semi-Automatic Lens Design by Electronic Computers. II. A Specific Computer Program," Journal of the Optical Society of America, Vol. 53, 1963, pp. 677-680.

5. Glatzel, E. "Ein neues Verfahren zur automatischen Korrektion optischer Systeme mit elektronischen Rechenmaschinen," Optik, Vol. 18, 1961, pp. 577-580.

6. Glatzel, E. and Wilson, R. "Adaptive Automatic Correction: Comparison of the Damped Least Squares Method and the Adaptive Method of Glatzel," Lens Design with Large Computers, ed. by W. L. Hyde. Rochester: Institute of Optics, 1966, pp. 1-47.

7. Glatzel, E. and Wilson, R. "Adaptive Automatic Correction in Optical Design," Applied Optics, Vol. 7, 1968, pp. 265-276.

8. Walling, D. D. Numerical Methods for Non-Linear Least Squares Curve Fitting. Ames: Iowa State University, UMF 64-3970, 1963.

9. Lesnick, J. P. and Rigler, A. K. Computer Curve Fitting of Spectral Absorptivity Data. Pittsburgh: Westinghouse Research Laboratories Ref. 67-9C1-LHICW-R3, AD 662-252, 1967.

10. von Holdt, R. E. "Inversion of Triple Diagonal Compound Matrices," Journal of the Association for Computing Machinery, Vol. 9, 1962, pp. 71-83.

11. Cornwell, L. W.; Pegis, R. J.; Rigler, A. K.; and Vogl, T. P. "Grey's Method for Nonlinear Optimization," Journal of the Optical Society of America, Vol. 63, 1973, pp. 576-581.

12. Davidon, W. C. "Variable Metric Method for Minimization," A. E. C. Research and Development Report, ANL-5990, 1959.

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

14. Marquardt, Donald W. "An Algorithm for Least-Squares Estimation of Non-Linear Parameters," Journal of the Society of Industrial Applied Mathematics, Vol. 2, 1963, pp. 164-168.

15. Hartley, H. O. "The Modified Gauss-Newton Method for Fitting of Non-Linear Regression Functions by Least Squares," Technometrics, Vol. 3, 1961, pp. 269-280.

16. St. Clair, D. C. and Rigler, A. K. "A Block Orthogonalization Method for Nonlinear Regression," Proceedings of the Computer Science and Statistics Eighth Annual Symposium on the Interface, ed. by J. Frane. Los Angeles: U.C.L.A. Health Sciences Computing Facility, 1975, pp. 280-287.

17. Powell, M. J. D. "A Survey of Numerical Methods for Unconstrained Optimization," SIAM Review, Vol. 12, 1970, pp. 79-97.

18. Powell, M. J. D. "Recent Advances in Unconstrained Optimization," Mathematical Programming Vol. I. Amsterdam: North-Holland Publishing Company, 1971, pp. 26-57.

19. Fletcher, R. "A Review of Methods for Unconstrained Optimization," Optimization, ed. by R. Fletcher. New York: Academic Press, 1969, pp. 1-12.

20. Dennis, J. E. "Some Computational Techniques for the Nonlinear Least Squares Problem," Numerical Solution of Systems of Nonlinear Algebraic Equations, ed. by G. Byrne and C. Hall. New York: Academic Press, 1973, pp. 157-183.

21. Broyden, C. G. "A Class of Methods for Solving Nonlinear Simultaneous Equations," Mathematics of Computation, Vol. 19, 1965, pp. 577-593.

22. Zeleznik, Frank J. "Quasi-Newton Methods for Nonlinear Equations," Journal of the Association for Computing Machinery, Vol. 15, 1968, pp. 256-271.

23. Huang, H. Y. "Unified Approach to Quadratically Convergent Algorithms for Function Minimization," Journal of Optimization Theory and Applications, Vol. 4, 1970, pp. 405-423.

24. Bass, R. "A Rank Two Algorithm for Unconstrained Minimization," Mathematics of Computation, Vol. 26, 1972, pp. 129-143.

25. Fletcher, R. "A New Approach to Variable Metric Algorithms," Computer Journal, Vol. 13, 1970, pp. 317-322.

26. Steen, N. M. and Byrne, G. D. "The Problem of Minimizing Nonlinear Functionals, I. Least Squares," Numerical Solution of Systems of Nonlinear Algebraic Equations, ed. by G. Byrne and C. Hall. New York: Academic Press, 1973, pp. 185-240.

27. Hooke, R. and Jeeves, T. A. "Direct Search Solution of Numerical and Statistical Problems," Journal of the Association for Computing Machinery, Vol. 8, 1961, pp. 212-229.

28. Himmelblau, D. M. Applied Nonlinear Programming. New York: McGraw-Hill, 1972.

29. Rosenbrock, H. H. "An Automatic Method for Finding the Greatest or Least Value of a Function," The Computer Journal, Vol. 3, 1960, pp. 175-184.

30. Swann, W. H. Report on the Development of a New Direct Search Method of Optimization. I. C. I. Ltd. Central Instrument Laboratory Research Note 64/3, 1964.

31. Lessman, R. E. Mixed Nonderivative Algorithms for Unconstrained Optimization. Rolla: University of Missouri-Rolla, 1975.

32. Cauchy, A. "Methods Generale Pour La Resolution Des Systems D` ´equations Simultanées," C. R. Acad. Sci. Paris, No. 25, 1847, pp. 368-381.

33. Goldstein, A. A. "Cauchy's Method of Minimization," *Numerische Mathematik,* Vol. 4, 1962, pp. 146-150.

34. Cragg, E. E. and Levy, A. V. "Study on a Supermemory Gradient Method for the Minimization of Functions," *Journal of Optimization Theory and Application,* Vol. 4, 1969, pp. 191-205.

35. Hestenes, M. R. and Stiefel, E. "Methods of Conjugate Gradients for Solving Linear Systems," *Journal of Research of the National Bureau of Standards,* Vol. 49, 1952, pp. 212-221.

36. Fletcher, R. and Reeves, C. M. "Function Minimization by Conjugate Gradients," *The Computer Journal,* Vol. 7, 1959, pp. 149-153.

37. Luenberger, D. G. "The Conjugate Residual Method for Constrained Minimization Problems," *SIAM Journal of Numerical Analysis,* Vol. 7, 1970, pp. 390-398.

38. Huang, H. Y. and Levy, A. V. "Numerical Experiments on Quadratically Convergent Algorithms for Function Minimization," *Journal of Optimization Theory and Application,* Vol. 6, 1970, pp. 269-282.

39. Zoutendijk, G. *Methods of Feasible Directions.* Amsterdam: Elsevier, 1960.

40. Wilde, D. J. and Beightler, C. S. *Foundations of Optimization.* Englewood Cliffs: Prentice-Hall, 1967, pp. 331-336.

41. Crockett, J. B. and Chernoff, H. "Gradient Methods of Maximization," *Pacific Journal of Mathematics,* Vol. 5, 1955, pp. 33-60.

42. Myers, G. E. "Properties of the Conjugate-Gradient and Davidon Methods," *Journal of Optimization Theory and Applications,* Vol. 2, 1968, pp. 209-219.

43. Stewart, G. W. "A Modification of Davidon's Minimization Method to Accept Difference Approximations of Derivatives," Journal of the Association for Computing Machinery, Vol. 4, 1962, pp. 441-452.

44. Lill, S. A. "A Modified Davidon Method for Finding the Minimum of a Function Using Difference Approximations for Derivatives," Computer Journal, Vol. 13, 1970, pp. 111-113.

45. McCormick, G.P. and Pearson, J. D. "Variable Metric Methods and Unconstrained Optimization," Optimization, ed. R. Fletcher. London: Academic Press, 1969, pp. 307-325.

46. Broyden, C. G. "Quasi-Newton Methods and Their Application to Function Minimization," Mathematics of Computation, Vol. 21, 1967, pp. 368-381.

47. Gauss, K. F. Theoria Motus Corporum Coelestiam. 1809.

48. Buchele, D. R. "Damping Factor for the Least Squares Method of Optical Design," Applied Optics, Vol. 7, pp. 2433-2435.

49. Tabata, T. and Ito, R. "Effective Treatment of the Interpolation Factor in Marquardt's Nonlinear Least-Squares Fit Algorithm," Computer Journal, Vol. 18, 1975, pp. 250-251.

50. Brown, K. M. and Dennis, J. E. Jr. "Derivative Free Analogues of the Levenberg-Marquardt and Gauss Algorithms," Numerische Mathematik, Vol. 18, 1972, pp. 289-297.

51. Meyer, R. R. "Theoretical and Computational Aspects of Nonlinear Regression," Nonlinear Programming, ed. by J. B. Rosen, O. L. Mangasarin, and K. Ritter. New York: Academic Press, 1970, pp. 465-485.

52. Smith, F. B. and Shanno, D. F. "An Improved Marquardt Procedure for Nonlinear Regression," Technometrics, Vol. 13, 1971, pp. 63-74.

53. Jones, A. "Spiral--A New Algorithm for Non-linear Parameter Estimation Using Least Squares," Computer Journal, Vol. 13, 1970, pp. 301-308.

54. Feder, D. P. "Lens Design Viewed as an Optimization Process," *Recent Advances in Optimization Techniques,* ed. by A. Lavi and T. P. Vogl. New York: John Wiley and Sons, 1965, pp. 5-21.

55. Pegis, R. J.; Grey, D. S.; Vogl, T. P.; and Rigler, A. K. "The Generalized Orthonormal Optimization Program and Its Application," *Recent Advances in Optimization Techniques,* ed. by A. Lavi and T. P. Vogl. New York: John Wiley and Sons, 1966.

56. Broste, N. A. *An Orthonormal Optimization Technique: Theory, Implementation and Application.* Pittsburgh: Carnegie-Mellon University, 1968.

57. Barham, R. H. *Parameter Reclassification in Nonlinear Least Squares.* Dallas: Southern Methodist University, UMF 72-648, 1971.

58. Barham, R. H. and Drane, Wanzer "An Algorithm for Least Squares Estimation of Nonlinear Parameters When Some of the Parameters are Linear," *Technometrics,* Vol. 14, 1972, pp. 757-766.

59. Ruhe, A. and Wedin, P. A. *Algorithms for Separable Nonlinear Least Squares Problems.* Stanford: Stanford University Report No. STAN-CS-74-434, 1974.

60. Golub, G. H. and Pereyra, V. "The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate," *SIAM Journal of Numerical Analysis,* Vol. 10, 1973, pp. 413-432.

61. Krough, F. T. "Efficient Implementation of a Variable Projection Algorithm for Nonlinear Least Squares Problems," *Communications of the A.C.M.,* Vol. 17, 1974, pp. 167-169.

62. Kaufman, L. "A Variable Projection Method for Solving Separable Nonlinear Least Squares Problems," *Bit,* Vol. 15, 1975, pp. 49-57.

63. Osborne, M. R. "Some Special Nonlinear Least Squares Problems," *SIAM Journal of Numerical Analysis,* Vol. 12, 1975, pp. 571-592.

64. Stone, John M. *Radiation and Optics.* New York: McGraw-Hill, Inc., 1963.

65. Mireles, R. "Determination of Parameters in Absorption Spectra by Numerical Minimization Techniques," *Journal of the Optical Society of America,* Vol. 56, 1966, pp. 644-647.

66. Zangwill, W. I. *Nonlinear Programming.* Englewood Cliffs: Prentice-Hall, 1969.

67. Murnaghan, F. D. *The Theory of Group Representations.* The Johns Hopkins Press, 1938, pp. 20-22.

68. Cornwell, L. W. *An Acceleration Technique for a Conjugate Direction Algorithm for Nonlinear Regression.* Rolla: University of Missouri-Rolla, 1972.

69. Aoki, M. *Introduction to Optimization Techniques.* New York: The Macmillan Company, 1971, pp. 148-150.

70. Mitchell, W. C. and McCraith, D. L. *Heuristic Analysis of Numerical Variants of the Gram-Schmidt Orthonormalization Process.* Stanford: Stanford University Report No. STAN-CS-122, 1969.

VITA

Daniel Clay St. Clair was born June 30, 1943, in Keokuk, Iowa, the son of Lee M. and Bessie Mae (Sharts) St. Clair. He received his secondary education at Canton High School in Canton, Missouri. In 1965, he received the Bachelor of Arts in Mathematics from Culver-Stockton College in Canton.

From 1965-1968, he was junior high mathematics instructor for Hamilton Community Consolidated Schools at Hamilton, Illinois, before being appointed high school mathematics instructor at Canton High School. From 1967-1969, he participated in sequential N.S.F. Summer Institutes at the College of William and Mary in Williamsburg, Virginia, receiving the Master of Teaching Science degree with a major in mathematics in 1969.

After serving as Assistant Professor of Mathematics and Director of the Computing Center at Culver-Stockton College from 1969-1971, he entered the University of Missouri-Rolla. While at the university, he attended two N.S.F. Computer Science Institutes and was recipient of a graduate assistantship and two Gulf Oil Corporation Fellowships. He is currently Assistant Professor of Mathematics and Computer Science at Western Kentucky University in Bowling Green, Kentucky.

He married the former Barbara Jean Barkley of Canton, Missouri, in June 1966. They have two sons; Joe and Clay.

APPENDIX A

Flow Chart of One-Dimensional Quadratic Fit*

In the flow chart below, the following variables and subprograms are used.

FVAL - Subprogram which determines the value of the objective function by using the vector in the argument.

DMIN1 - Function subprogram which determines the minimum of the scalar values in the argument list.

ALPHA - Vector of $\vec{\alpha}$-values from the algorithm.

DELALP - Direction vector determined by the algorithm.

B, C - Multiples of DELALP used for the quadratic fit.

SAVE - Maximum value allowable for B. One tenth proved to be an acceptable value for the problems tested.

XMIN - Multiple of DELALP at which the predicted minimum function value, FCTXM, is located.

SECDER - Value of the second derivative of the quadratic at the predicted minimum point ALPHA + XMIN*DELALP.

M - Scalar used to limit the size of XMIN, i.e. XMIN would not be accepted as a step length

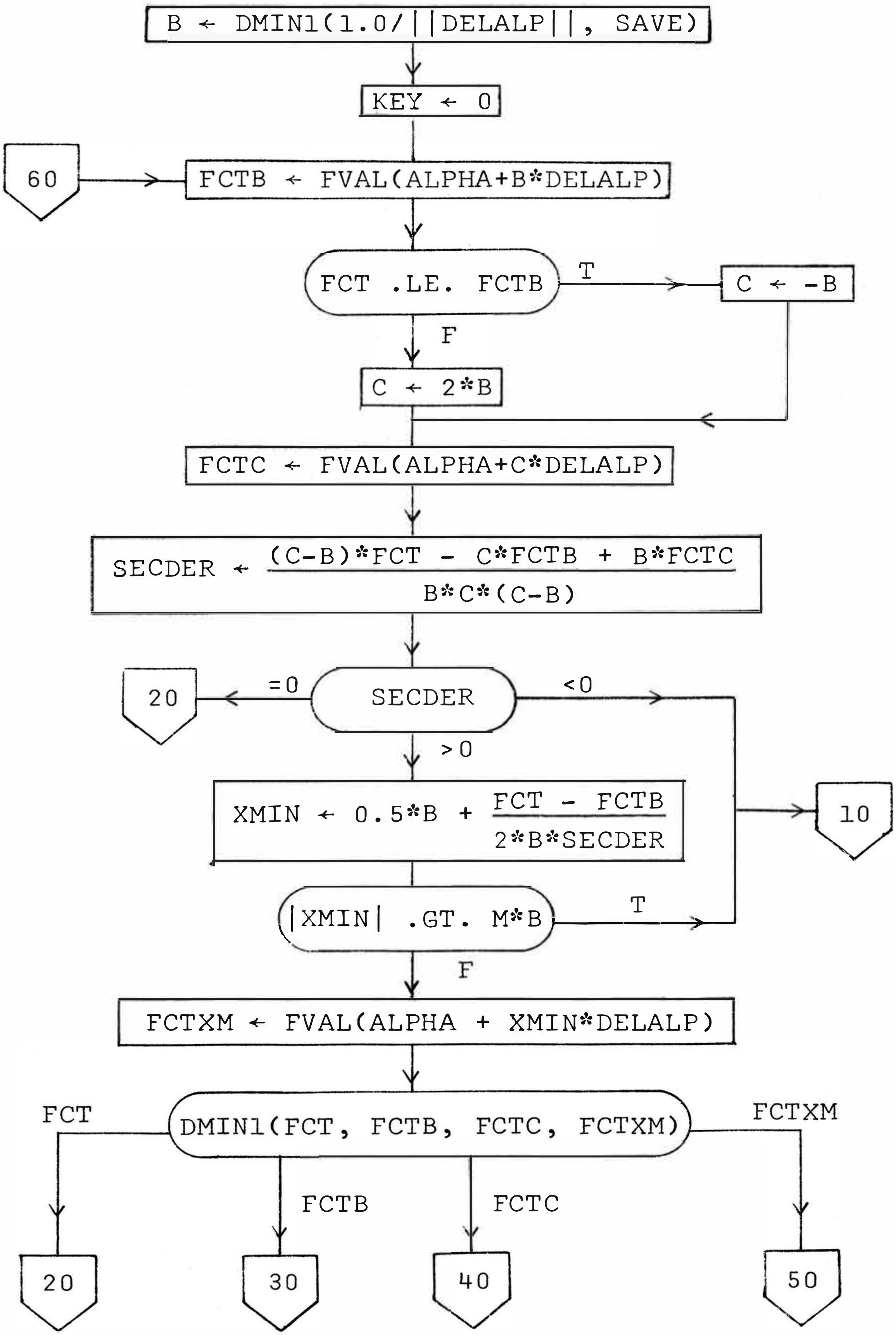
*This algorithm is a modification of the one presented by Aoki [69].

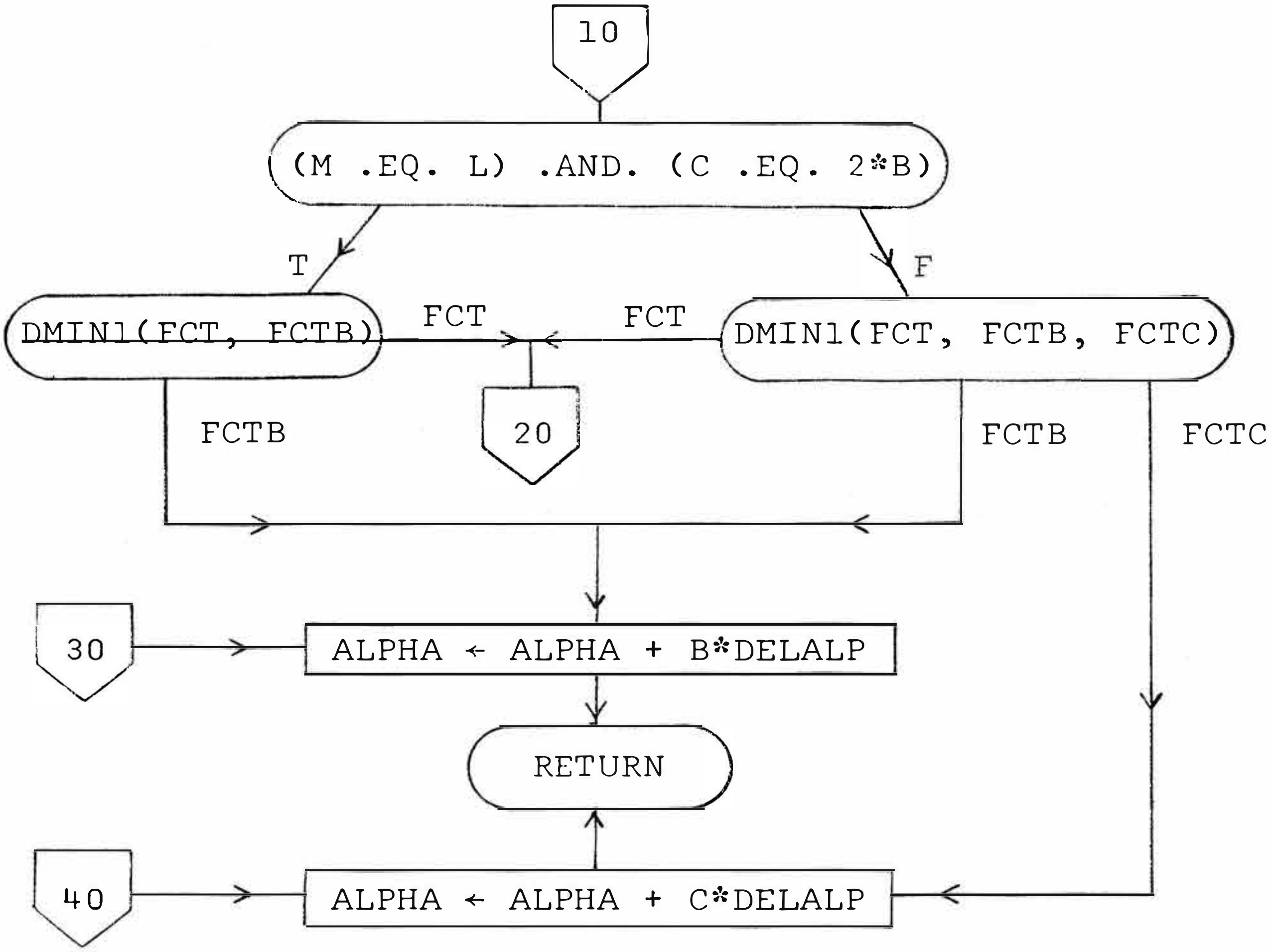
if |XMIN| > B*M. In the problems tested, ten proved to be a satisfactory value for M.

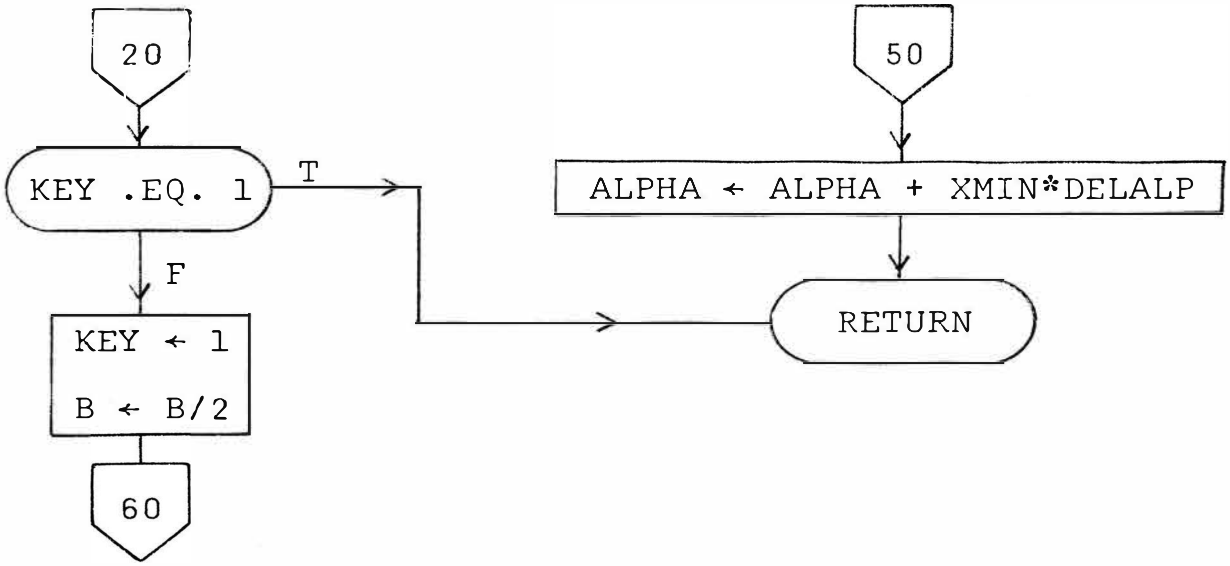
FCTB - Value of objective function at ALPHA + B*DELALP.

FCTC - Value of objective function at ALPHA + C*DELALP.

FCTXM - Value of objective function at the predicted minimum point, ALPHA + XMIN*DELALP.


B ← DMIN1(1.0/||DELALP||, SAVE)
KEY ← 0
60
FCTB ← FVAL(ALPHA+B*DELALP)
FCT .LE. FCTB
T
C ← -B
F
C ← 2*B
FCTC ← FVAL(ALPHA+C*DELALP)
SECDER ← ((C-B)*FCT - C*FCTB + B*FCTC) / (B*C*(C-B))
20
=0
SECDER
<0
>0
XMIN ← 0.5*B + (FCT - FCTB) / (2*B*SECDER)
10
|XMIN| .GT. M*B
T
F
FCTXM ← FVAL(ALPHA + XMIN*DELALP)
FCT
DMIN1(FCT, FCTB, FCTC, FCTXM)
FCTXM
FCTB
FCTC
20
30
40
50


10
(M .EQ. L) .AND. (C .EQ. 2*B)
T
F
DMIN1(FCT, FCTB)
FCT
FCT
DMIN1(FCT, FCTB, FCTC)
FCTB
20
FCTB
FCTC
30
ALPHA ← ALPHA + B*DELALP
RETURN
40
ALPHA ← ALPHA + C*DELALP


20
KEY .EQ. 1
T
F
KEY ← 1
B ← B/2
60
50
ALPHA ← ALPHA + XMIN*DELALP
RETURN

APPENDIX B

Conjugate Direction Theorems

This appendix is devoted to the proofs of the conjugate direction theorems presented in Chapter II. The definitions, theorems, and proofs presented here are well-known in the literature.

DEFINITION 2.1 The quadratic function

$$g(\vec{\alpha}) = a + \vec{c}^T\vec{\alpha} + 1/2\ \vec{\alpha}^TQ\vec{\alpha} \qquad (2.6)$$

defined on R^m is said to be positive definite provided

$$\vec{\alpha}^TQ\vec{\alpha} > 0 \quad \text{for all } \vec{\alpha} \neq \vec{0}.$$

DEFINITION 2.2 Given the positive definite quadratic function

$$g(\vec{\alpha}) = a + \vec{c}^T\vec{\alpha} + 1/2\ \vec{\alpha}^TQ\vec{\alpha}, \qquad (2.6)\text{bis}$$

defined on R^m, any two vectors $\vec{p},\vec{q} \in R^m$ are said to be conjugate with respect to Q provided they are nonzero and

$$\vec{p}^TQ\vec{q} = 0. \qquad (2.7)$$

Any set of vectors $\{\vec{p}_i\} \subset R^m$ for which $\vec{p}_i^TQ\vec{p}_j = 0$ when $i \neq j$ is said to be mutually conjugate with respect to Q.

THEOREM 2.1 If $\{\vec{p}_i\} \subset R^m$ is a set of vectors which are mutually conjugate with respect to matrix Q of equation (2.6), then $\{\vec{p}_i\}$ is a linearly independent set.

PROOF Let $\{\vec{p}_i\} \subset R^m$ be a set of k vectors which are mutually conjugate with respect to the positive definite matrix Q of equation (2.6).

To show $\{\vec{p}_i\}$ is a linearly independent set it will be shown that for any linear combination of $\{\vec{p}_i\}$ such that

$$d_1\vec{p}_1 + \ldots + d_i\vec{p}_i + \ldots + d_k\vec{p}_k = 0$$

then each of the scalars $d_t = 0$ for $t = 1,2,\ldots,k$.

Consider the scalar d_i. Multiplying the above expression by $\vec{p}_i^{\,T}Q$ gives

$$d_1\vec{p}_i^{\,T}Q\vec{p}_1 + \ldots + d_i\vec{p}_i^{\,T}Q\vec{p}_i + \ldots + d_k\vec{p}_i^{\,T}Q\vec{p}_k = 0$$

which reduces to

$$d_i\vec{p}_i^{\,T}Q\vec{p}_i = 0.$$

Now Q is positive definite; therefore, $\vec{p}_i^{\,T}Q\vec{p}_i \neq 0$ since $\vec{p}_i \neq \vec{0}$ by definition. It must be the case that $d_i = 0$. Therefore $d_t = 0$ for $t = 1,2,\ldots,k$ and $\{\vec{p}_i\}$ must be a linearly independent set.

Q.E.D.

THEOREM 2.2 If $\{\vec{p}_i\} \subset R^m$ is a set of m mutually conjugate vectors with respect to Q, then the positive definite quadratic of equation (2.6) can be minimized by sequentially minimizing in each of the directions $\vec{p}_1,\ldots,\vec{p}_m$ exactly once.

PROOF Let $\{\vec{p}_i\} \subset R^m$ be a set of m mutually conjugate vectors with respect to matrix Q of equation (2.6). By Theorem 2.1, $\{\vec{p}_i\}$ is a linearly independent set of vectors. Since this set of vectors also spans R^m, it forms a basis of R^m.

If $\vec{\alpha}*$ is the minimum of $g(\vec{\alpha})$ in equation (2.6), one may write $\vec{\alpha}*$ as a linear combination of the basis vectors, i.e.

$$\vec{\alpha}* = \sum_{i=1}^{m} d_i\vec{p}_i.$$

Substituting into $g(\vec{\alpha})$

$$\begin{aligned} g(\vec{\alpha}*) &= g(\sum_{i=1}^{m} d_i\vec{p}_i) \\ &= a + \vec{c}^T(\sum_{i=1}^{m} d_i\vec{p}_i) + (\sum_{i=1}^{m} d_i\vec{p}_i^T)Q(\sum_{i=1}^{m} d_i\vec{p}_i) \\ &= a + \vec{c}^T(\sum_{i=1}^{m} d_i\vec{p}_i) + \sum_{i=1}^{m} d_i^2(\vec{p}_i^TQ\vec{p}_i) \end{aligned}$$

where the simplification in the last expression comes from the Q-conjugacy of $\{\vec{p}_i\}$.

Since each term in the last expression depends only on one direction $\vec{p}_i$ and one parameter d_i, m independent minimizations in the directions $\vec{p}_1,\ldots,\vec{p}_m$ will determine d_i, $i = 1,2,\ldots,m$ and will therefore determine $\vec{\alpha}^*$.

Q.E.D.

APPENDIX C

Flow Chart of Cornwell's Spacer Step*

In the flow chart below, the following variables and subprogram are used.

FVAL - Subprogram which determines the value of the objective function by using the vector in the argument.

FCT - Minimum value of the objective function.

FCTF - Current value of the objective function.

ALPHA - Vector of current $\vec{\alpha}$-values.

ALHOLD - Vector of $\vec{\alpha}$-values at the beginning of the conjugate direction algorithm.

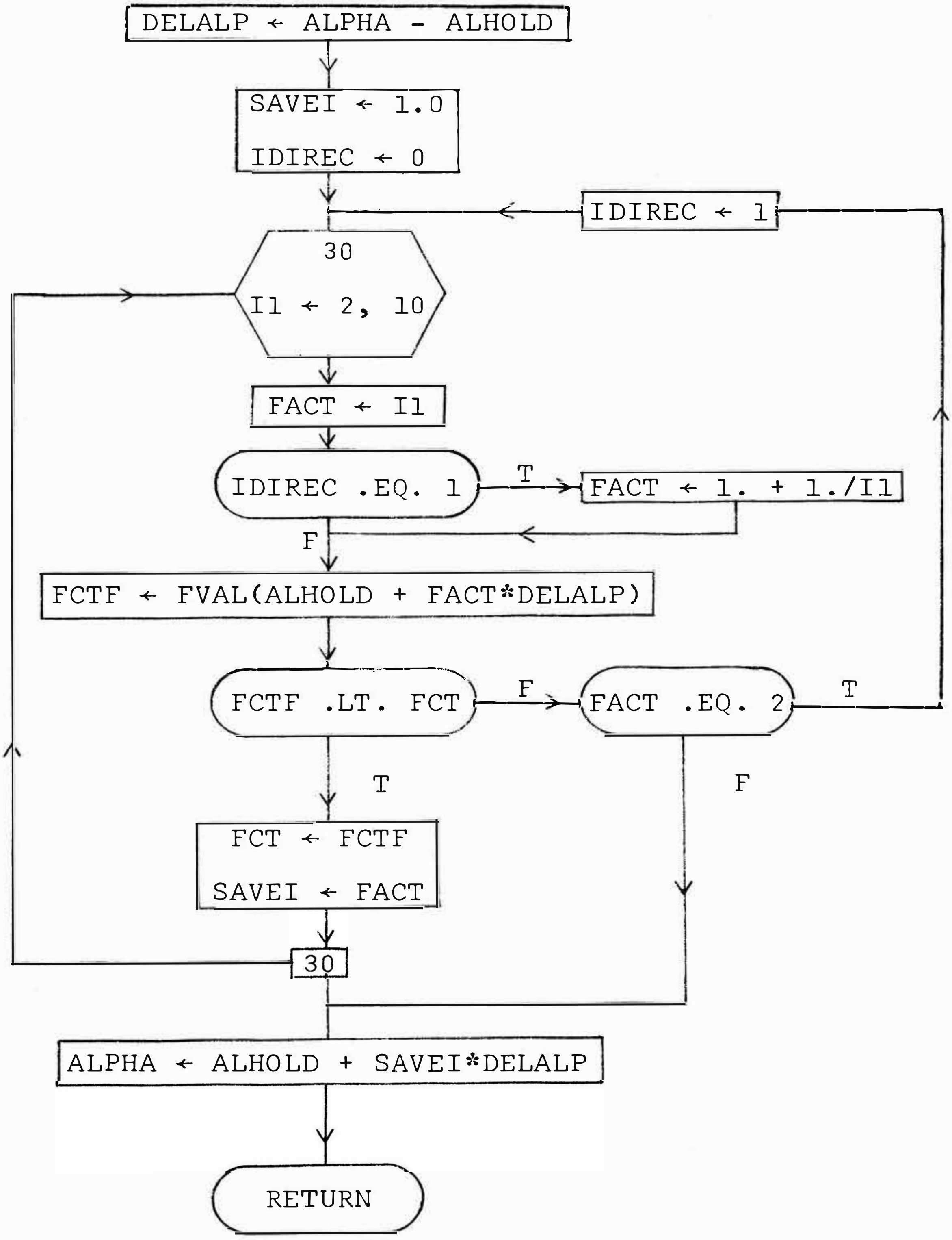
DELALP - Direction vector determined by the algorithm.

IDIREC - Flag. If IDIREC = 0, search is adding multiples of DELALP. If IDIREC = 1, search is adding increments of DELALP.

FACT - Amount of DELALP being added to base point ALHOLD.

SAVEI - Holds last successful value of FACT.

*This flow chart was suggested in Cornwell's dissertation [68].


DELALP ← ALPHA - ALHOLD
SAVEI ← 1.0
IDIREC ← 0
IDIREC ← 1
30
I1 ← 2, 10
FACT ← I1
IDIREC .EQ. 1
T
FACT ← 1. + 1./I1
F
FCTF ← FVAL(ALHOLD + FACT*DELALP)
FCTF .LT. FCT
F
FACT .EQ. 2
T
T
F
FCT ← FCTF
SAVEI ← FACT
30
ALPHA ← ALHOLD + SAVEI*DELALP
RETURN

APPENDIX D

Cholesky Decomposition Algorithm

Let A be a real, symmetric, positive definite matrix of order m. Cholesky decomposition can be used to factor A into the product

$$A = R^T R$$

where R is a unique upper triangular matrix whose elements are computed by the following relations.

$$r_{11} = (a_{11})^{1/2}$$

$$r_{1j} = a_{1j}/r_{11} \qquad j = 2,\ldots,m.$$

For i = 2,...,m

$$r_{ii} = (a_{ii} - \sum_{k=1}^{i-1} r_{ki}^2)^{1/2}$$

$$r_{ij} = (a_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj})/r_{ii}$$

$$j = i+1,\ldots,m.$$

APPENDIX E

Comparison of the Classical and Blocked Gram-Schmidt Processes

The results of this appendix verify that the classical and blocked Gram-Schmidt algorithms described in Chapter III produce identical matrix factorizations when applied to a constant matrix H.

Let A be an nxm matrix which has been blocked into m/j column matrices each containing j columns. The following notations will be used in this appendix in reference to any such matrix A.

a_{ik} - the scalar in row i column k of A.

$\vec{A}_i$ - the i-th column of A.

A_i - the i-th column block of A. This nxj matrix consists of columns (i-1)j+1 - ij of A.

A_{ik} - the jxj matrix consisting of rows (i-1)j+1 - ij and columns (k-1)j+1 - kj of A. Note that $a_{st} \varepsilon A_{ik}$ provided $s = (i-1)j+u$ and $t = (k-1)j+w$ where $1 \leq u,w \leq j$.

A(i) - the nxi matrix consisting of the first i columns of A. Note that A = A(m).

A(i,k) - the ixk matrix consisting of the first i rows and k columns of A. Note that A = A(n,m).

Let H be a constant matrix of n rows and m linearly independent columns. The classical Gram-Schmidt process factors H into the matrix product H = GB where $G^TG = I$ and B is a nonsingular, upper triangular matrix. The classical Gram-Schmidt algorithm is applied to each column of H as shown.

$$\vec{G}_1 = \vec{H}_1/||\vec{H}_1|| \tag{3.4)bis}$$

$$\vec{D}_1 = \vec{H}_i - \sum_{k=1}^{i-1} \vec{G}_k b_{ki} \tag{3.5)bis}$$

$$\vec{G}_i = \vec{D}_i/||\vec{D}_i|| \qquad i = 2,3,\ldots,m \tag{3.6)bis}$$

where

$$b_{ki} = \vec{G}_k^T\vec{H}_i \tag{3.7)bis}$$

$$||\vec{D}_i|| = (\vec{D}_i^T\vec{D}_i)^{1/2} \tag{3.8)bis}$$

The following identity is obtained directly from equations (3.4-3.8).

$$b_{ii} = ||\vec{D}_i|| = [\vec{H}_i^T\vec{H}_i - \sum_{k=1}^{i-1} b_{ki}^2]^{1/2} \tag{E.1}$$

Further, since G and B are formed in a stepwise manner and B is upper triangular, at the end of the i-th step of the process

$$H(i) = G(i)B(i,i). \qquad (E.2)$$

The blocked Gram-Schmidt process can also be applied to matrix H to produce H = QR where $Q^TQ = I$ and B is a nonsingular, upper triangular matrix. Unlike the classical algorithm, the blocked algorithm factors j columns of H at each step.

$$Q_1 = H_1L_1^{-1} \qquad (E.3)$$

$$F_i = H_i - \sum_{k=1}^{i-1} Q_kR_{ki} \qquad (E.4)$$

$$Q_i = F_iL_i^{-1} \qquad i = 2,3,\ldots,m/j \qquad (E.5)$$

where

$$R_{ki} = Q_k^TH_i \qquad (E.6)$$

$$F_i^TF_i = L_i^TL_i \qquad (E.7)$$

The Cholesky decomposition algorithm described in Appendix D is used to produce the nonsingular, upper triangular matrix L_i. (See Theorem 3.1 for the proof that L_i is nonsingular when the columns of H are linearly independent.) The matrices Q_i, F_i, and H_i

are nxj while R_{ki} and L_i are jxj. From the notation introduced at the beginning of this appendix, one can think of F_i and R_{ki} as being members of matrices F and R respectively. Note that $R_{ii} = L_i$.

Since H is factored in a stepwise manner in blocks of j columns each and R is upper triangular, at the i-th step

$$H(ij) = Q(ij)R(ij,ij) \quad \text{for } i = 1,2,\ldots,m/j. \qquad (E.8)$$

If $F = [F_1,F_2,\ldots,F_{m/j}]$ then

$$F^TF = \begin{bmatrix} F_1^TF_1 & & & & \cdot\ \cdot\ 0 \\ & \cdot & & & \vdots \\ & & F_i^TF_i & & \\ \vdots & & & \cdot & \\ 0\ \cdot\ \cdot & & & & F_{m/j}^TF_{m/j} \end{bmatrix}$$

where F_i is the nxj column matrix from equation (E.4). Let f_{ks} represent the scalar in row k column s of F^TF. If k and s are of the forms $k = (i-1)j+p$ and $s = (i-1)j+t$ where $1 \leq p,t \leq j$ then f_{ks} is an element in $F_i^TF_i$ of F^TF. To find an expression for f_{ks}, consider the identity

$$F_i^TF_i = H_i^TH_i - \sum_{u=1}^{i-1} R_{ui}^TR_{ui}. \qquad (E.9)$$

If r_{ks} ε R, then r_{ks} ε R_{ui} provided k = (u-1)j+p and s = (i-1)j+t for 1 ≤ p,t ≤ j. Using equation (E.9), f_{ks} can be expressed as

$$f_{ks} = \vec{H}_k^T\vec{H}_s - \sum_{u=1}^{i-1}\sum_{w=1}^{j} r_{(u-1)j+w\ k}\, r_{(u-1)j+w\ s}$$

or

$$f_{ks} = \vec{H}_k^T\vec{H}_s - \sum_{u=1}^{(i-1)j} r_{uk}\, r_{us} \qquad \text{(E.10)}$$

where k = (i-1)j+p and s = (i-1)j+t for 1 ≤ p,t ≤ j. Therefore, equation (E.10) provides a way of expressing the elements in the diagonal blocks of $F^T\mathbf{F}$ in terms of elements from H and R.

Theorem E.1 verifies that the results of factoring matrix H using the classical Gram-Schmidt process are identical to those produced by factoring H with the blocked Gram-Schmidt algorithm.

THEOREM E.1 Let H be a constant matrix with n rows and m columns such that the columns of H are linearly independent. Further, let H be factored by the classical Gram-Schmidt process into H = GB and by the blocked Gram-Schmidt process into H = QR where G and Q are the orthonormal matrices produced by each of the respective methods. B and R are the corresponding upper Gram-Schmidt transformation matrices. Then G = Q and B = **R**.

PROOF Using the notations and results established earlier in this appendix, the theorem will be proven by inducting on the column blocks of Q and R. Without loss of generality, it will be assumed the blocked Gram-Schmidt process is applied to blocks of equal size, i.e. say j columns per block.

First, it is necessary to show $R_1 = B_1$, i.e. $R_{11} = B_{11}$. Recall that $R_{11} = L_1$ where L_1 comes from the Cholesky decomposition of $F_1^T F_1$. The proof will be established by inducting on the rows of R_{11}. It will first be shown that $r_{1s} = b_{1s}$ for s = 1,2,...,j.

Using the Cholesky decomposition algorithm of Appendix D on $F_1^T F_1$ and noting that i = p = t = 1 in equation (E.10)

$$r_{11} = (f_{11})^{1/2} = ||\vec{H}_1|| = b_{11}.$$

Now, assume $r_{1s} = b_{1s}$ for $s = 1,2,\ldots,t-1<j$. It will be shown that $r_{1t} = b_{1t}$. From the Cholesky decomposition algorithm, the preceeding assumption, and equations (3.4, E.10)

$$r_{1t} = f_{1t}/r_{11} = (\vec{H}_1^T \vec{H}_t)/b_{11}$$

$$= \vec{G}_1^T \vec{H}_t = b_{1t}.$$

Hence, $r_{1s} = b_{1s}$ for s = 1,2,...,j.

Now, assume the first k-1 rows of R_{11} are equal to the first k-1 rows of B_{11} where $k-1<j$. Induction will again be applied to show the k-th rows are equal. Since B and R are upper triangular, the induction begins at r_{kk}. Using Cholesky decomposition, equation (E.10) with $i = 1$, $p = t = k$, the inductive assumption on the first k-1 rows of B_{11}, and equation (E.1)

$$r_{kk} = (f_{kk} - \sum_{u=1}^{k-1} r_{uk}^2)^{1/2}$$

$$= (\vec{H}_k^T\vec{H}_k - \sum_{u=1}^{k-1} b_{uk}^2)^{1/2}$$

$$= b_{kk}.$$

To complete the induction on the k-th row of R_{11}, assume $r_{ks} = b_{ks}$ for $s = k,\ldots,t-1<j$. It will be shown that $r_{kt} = b_{kt}$. Using Cholesky decomposition, equation (E.10) where $i = 1$, $p = k$, and the inductive assumptions on the first k-1 rows of R_{11} and the first t-1 columns of the k-th row

$$r_{kt} = (f_{kt} - \sum_{u=1}^{k-1} r_{uk}r_{ut})/r_{kk}$$

$$= (\vec{H}_k^T\vec{H}_t - \sum_{u=1}^{k-1} b_{uk}b_{ut})/b_{kk}$$

$$= (\vec{H}_k - \sum_{u=1}^{k-1} b_{uk}\vec{G}_u)^T\vec{H}_t/b_{kk}$$

$$r_{kt} = \vec{G}_k^T \vec{H}_t = b_{kt}$$

where the last row in the above argument was obtained using equations (3.4-3.8). This completes the induction showing $r_{ks} = b_{ks}$ for $s = 1,2,\ldots,j$. This also completes the induction on the rows of R_{11}. Hence, $R_{11} = B_{11}$.

Since the columns of H are linearly independent, Theorem 3.1 guarantees that R and hence R_{11} are nonsingular. Using this fact and equations (E.2, E.8)

$$H(j) = G(j)B(j,j) = Q(j)R(j,j)$$

or

$$G(j) = Q(j)$$

since $B(j,j) = R(j,j) = R_{11}$. Hence, the first j columns of G and Q are identical.

To complete the induction on the column blocks of R, assume $B(uj) = R(uj)$ for $u = 1,2,\ldots,i-1$. Then equations (E.2, E.8) imply that $G(uj) = Q(uj)$ for $u = 1,2,\ldots,i-1$. To show that the i-th column block of R is identical to the i-th column block of B, i.e. $R_i = B_i$, it will be shown that $R_{ki} = B_{ki}$ for $k = 1,2,$ $\ldots,i-1$. Next it will be shown that $R_{ii} = B_{ii}$. The upper triangular form of R and B give $R_{ki} = B_{ki} = 0$ for $k = i+1,\ldots,j$.

To show $R_{ki} = B_{ki}$ for $k = 1,2,\ldots,i-1$, let r_{vw} be an element of matrix R. This element will be in R_{ki} provided $v = (k-1)j+p$ and $w = (i-1)j+t$ where $1 \leq p, t \leq j$. Then from equation (E.6)

$$r_{vw} = \vec{Q}_v^T\vec{H}_w = \vec{Q}^T_{(k-1)j+p}\vec{H}_w$$

$$= \vec{G}_v^T\vec{H}_w = b_{vw}$$

since by the inductive assumption $\vec{Q}_{(k-1)j+p} = \vec{G}_{(k-1)j+p}$ for $k = 1,2,\ldots,i-1$ and $p = 1,2,\ldots,j$. Therefore $R_{ki} = B_{ki}$ for $k = 1,2,\ldots,i-1$.

Induction will be used on the rows of R_{ii} to show $R_{ii} = B_{ii}$. As equations (E.5-E.7) of the blocked Gram-Schmidt process verify, $R_{ii} = L_i$ where L_i is obtained from the Cholesky decomposition of $F_i^TF_i$. If f_{ks} is an element of F^TF then f_{ks} is an element of $F_i^TF_i$ provided $k = (i-1)j+p$ and $s = (i-1)j+t$ where $1 \leq p, t \leq j$.

To show the first row of R_{ii} is equal to the first row of B_{ii}, i.e. $r_{(i-1)j+1\ w} = b_{(i-1)j+1\ w}$ for $w = (i-1)j+1,\ldots,ij$, induction is performed on w. Hence, for $w = (i-1)j+1$ the Cholesky decomposition algorithm is used to derive the element in the first row and first column of $R_{ii} = L_i$. Equations (E.1, E.10) and the fact that $r_{kw} = b_{kw}$ for $k = 1,\ldots,w-1$ are used to show

$$\begin{aligned} r_{ww} &= (f_{ww})^{1/2} \\ &= (\vec{H}_w^T\vec{H}_w - \sum_{u=1}^{w-1} r_{uw}^2)^{1/2} \\ &= (\vec{H}_w^T\vec{H}_w - \sum_{u=1}^{w-1} b_{uw}^2)^{1/2} \\ &= b_{ww}. \end{aligned}$$

Next, assume $r_{ws} = b_{ws}$ for $s = w,\ldots,t-1<j$ where $w = (i-1)j+1$. It must be shown that $r_{wt} = b_{wt}$. Using Cholesky decomposition, equations (3.5-3.7, E.10), and the inductive assumption

$$\begin{aligned} r_{wt} &= f_{wt}/r_{ww} \\ &= (\vec{H}_w^T\vec{H}_t - \sum_{u=1}^{w-1} r_{uw}r_{ut})/r_{ww} \\ &= \vec{G}_w^T\vec{H}_t = b_{wt}. \end{aligned}$$

This completes the induction on the first row of R_{ii}.

Assume the first k-1 rows of R_{ii} are equal to the first k-1 rows of B_{ii}, i.e. $r_{(i-1)j+p\ w} = b_{(i-1)j+p\ w}$ for $p = 1,2,\ldots,k-1<j$, $w = (i-1)j+1,\ldots,ij$. Induction will be used to show $r_{vw} = b_{vw}$ where $v = (i-1)j+k$ and $w = (i-1)j+1,\ldots,ij$. Note that $r_{vw} = b_{vw} = 0$ for $w = (i-1)j+1,\ldots,v-1$. By Cholesky decomposition, the

inductive assumption on the rows of R_{ii}, and equations (E.1, E.10)

$$r_{vv} = (f_{vv} - \sum_{u=1}^{k-1} r^2_{s+u\ v})^{1/2}$$

$$= (\vec{H}^T_v\vec{H}_v - \sum_{u=1}^{s} r^2_{uv} - \sum_{u=s+1}^{v-1} r^2_{uv})^{1/2}$$

$$= (\vec{H}^T_v\vec{H}_v - \sum_{u=1}^{v-1} r^2_{uv})^{1/2}$$

$$= b_{vv}$$

where s = (i-1)j.

To complete the induction on the k-th row of R_{ii}, assume $r_{vt} = b_{vt}$ for v = (i-1)j+k and t = (i-1)j+1, ...,w-1<j. It must be shown that $r_{vw} = b_{vw}$. By Cholesky decomposition, the inductive assumptions and equations (3.5-3.7, E.10)

$$r_{vw} = (f_{vw} - \sum_{u=1}^{k-1} r_{s+u\ v}r_{s+u\ w})/r_{vv}$$

$$= (\vec{H}^T_v\vec{H}_w - \sum_{u=s+1}^{v-1} r_{uv}r_{uw})/r_{vv}$$

$$= \vec{G}^T_v\vec{H}_w = b_{vw}$$

where $s = (i-1)j$. This completes the induction on the k-th row of R_{ii}. Hence, $R_{ii} = B_{ii}$. Further, this completes the induction on R_i showing that $R_i = B_i$. Then, from equations (E.2, E.8), $Q(ij) = G(ij)$.

It has been shown that $R_1 = B_1$ and $Q_1 = G_1$. Further, assuming $R(uj) = B(uj)$ for $u = 1,2,\ldots,i-1$, it has been shown that $Q(uj) = G(uj)$, $R(ij) = B(ij)$, and $Q(ij) = B(ij)$. Hence, by induction, it must be the case that $R = B$ and $Q = G$.

Q.E.D.

The results of Theorem E.1 provide the proof to the following corollary.

COROLLARY E.1 Let Q be the matrix described in Theorem E.1 which is produced by the blocked Gram-Schmidt algorithm. Then

$$Q^TQ = I.$$

APPENDIX F

Convergence of Cornwell's GOOP-LAT Algorithm

The mathematical framework presented in this appendix was developed by Zangwill [57] and used by Cornwell [68] in proving convergence of the GOOP-LAT algorithm. Proofs of Theorems F.1 and F.2 may be found in Zangwill's work. Cornwell proves Theorems F.3 and F.4.

DEFINITION F.1 By a point-to-set map, it is meant that for any point $\vec{\alpha} \varepsilon R^m$, $A(\vec{\alpha})$ is a set in R^m, i.e. $A : R^m \rightarrow R^m$.

DEFINITION F.2 An algorithm is an iterative process consisting of a sequence of point-to-set maps $A_k : R^m \rightarrow R^m$. Given a point $\vec{\alpha}_1$, a sequence of points $\{\vec{\alpha}_k\}$ is generated recursively by

$$\vec{\alpha}_{k+1} \varepsilon A_k(\vec{\alpha}_k)$$

where any point in the set $A_k(\vec{\alpha}_k)$ is a possible successor point $\vec{\alpha}_{k+1}$.

DEFINITION F.3 A solution set is the set of all optimal points and a solution point is a point in Ω, the solution set.

DEFINITION F.4 A point-to-set map $A : R^m \rightarrow R^m$ is closed at the limit point $\vec{\alpha}_\infty$

if $\qquad \vec{\alpha}_k \to \vec{\alpha}_\infty, \ \vec{\beta}_k \ \varepsilon \ A(\vec{\alpha}_k)$, and $\vec{\beta}_k \to \vec{\beta}_\infty$

for $\qquad k \ \varepsilon \ \kappa$ implies $\vec{\beta}_\infty \ \varepsilon \ A(\vec{\alpha}_\infty)$.

The map is closed on $X \subset R^m$ if it is closed at each $\vec{\alpha} \ \varepsilon \ X$.

<u>THEOREM F.1</u> Let the point-to-set map $A : R^m \to R^m$ determine an algorithm that given a point $\vec{\alpha}_1 \ \varepsilon \ R^m$ generates the sequence $\{\vec{\alpha}_k\}$. Let a solution set $\Omega \subset R^m$ be given.

Suppose

1. All points $\vec{\alpha}_k$ are in a compact set $X \subset R^m$.
2. There exists a continuous function $g : R^m \to R^1$ such that

   a.) if $\vec{\alpha} \not\varepsilon \ \Omega$, then for any $\vec{\beta} \ \varepsilon \ A(\vec{\alpha})$

   $$g(\vec{\beta}) < g(\vec{\alpha}),$$

   b.) if $\vec{\alpha} \ \varepsilon \ \Omega$, then either the algorithm terminates or for any $\vec{\beta} \ \varepsilon \ A(\vec{\alpha})$

   $$g(\vec{\beta}) \leq g(\vec{\alpha}),$$

   and
3. The map A is closed at $\vec{\alpha} \not\varepsilon \ \Omega$.

Then either the algorithm stops at a solution, or the limit of any convergent subsequence is a solution.

DEFINITION F.5 A mixed algorithm is an algorithm that has a given basic algorithm map B, which depends only upon $\vec{\alpha}$, such that

$$B = A_k \qquad k \varepsilon \kappa.$$

In other words, the basic map B is used infinitely often. Other maps are employed for the remaining k.

THEOREM F.2 Let B : $R^m \rightarrow R^m$ be an algorithmic map for nonlinear programming problems such that B satisfies conditions 1, 2, and 3 of Theorem F.1. Let a mixed algorithm for the problem be defined by the maps A_k : $R^m \rightarrow R^m$ such that for some κ

$$A_k = B \qquad k \varepsilon \kappa$$

while

$$g(\vec{\alpha}_{k+1}) \leq g(\vec{\alpha}_k) \quad \text{for } k \not\varepsilon \kappa.$$

Assume that

1. All $\vec{\alpha}_k$ ε X where X is compact, and
2. If $\vec{\alpha}^*$ ε Ω, and

$$g(\vec{\beta}) \leq g(\vec{\alpha}^*)$$

then

$$\vec{\beta} \varepsilon \Omega.$$

Then under these hypotheses the mixed algorithm either stops at a solution or generates a sequence $\{\vec{\alpha}_k\}$ such that the limit of any convergent subsequence is a solution point.

<u>DEFINITION F.6</u> Let $M^1 : R^{2m} \to R^m$ represent a one-dimensional search algorithm where M^1 has the form

$$M^1(\vec{\alpha},\Delta\vec{\alpha}) = \{\vec{\beta} \mid g(\vec{\beta}) = \min_{\tau \varepsilon J} g(\vec{\alpha} + \tau\Delta\vec{\alpha}),\ \vec{\beta} = \vec{\alpha} + \tau_0\Delta\vec{\alpha}\}.$$

J is an interval over which the scalar τ varies and $(\vec{\alpha},\Delta\vec{\alpha}) \ \varepsilon\ R^{2m}$.

<u>THEOREM F.3</u> Let the objective function defined on the compact set X be continuous and have a unique minimum. The algorithmic map $A = M^1D$ where M^1 is the one-dimensional search of Definition F.6 and $D(\vec{\alpha}) = (\vec{\alpha},\vec{\gamma})$, $\vec{\gamma} = \vec{\alpha}_{k(m+1)} - \vec{\alpha}_k$, is convergent. The interval for M^1 is $J = [-\rho,\rho]$.

<u>THEOREM F.4</u> Using the assumptions of Theorem F.3, Cornwell's GOOP-LAT algorithm, a mixed algorithm using the algorithmic map $A = M^1D$ as a spacer step, is convergent.

APPENDIX G

Details of Algorithm Implementations

Details pertinent to the algorithm implementations used in the numerical results recorded in Chapter IV are given in this appendix.

1. Davidon-Fletcher-Powell (DFP)

The Davidon-Fletcher-Powell (DFP) algorithm belongs to the broad class of General Methods discussed in Chapter II. Although it was the only method tested from this class, it has the reputation of being one of the more successful algorithms in its class.

Since the method searches in conjugate directions by estimating the inverse Hessian matrix, it required an initial approximation of this matrix along with initial parameter estimates. The present implementation used the mxm identity matrix as the initial approximation of the inverse Hessian. As suggested by McCormick and Pearson [45], the inverse Hessian approximation was reset to the identity after every m steps.

Fletcher and Powell [13] indicate that the best performance of the DFP algorithm is achieved when a rather accurate one-dimensional search is used to determine the step length in each of the conjugate directions chosen. The present implementation used the one-dimensional cubic search from their paper.

2. Levenberg-Marquardt (LM)

The Levenberg-Marquardt (LM) algorithm, as well as the remaining methods tested, belongs to the class of Least Squares Methods. The LM algorithm varies from the other algorithms tested in that step length and direction are chosen simultaneously.

The algorithm required the initialization of the damping factor λ (see equation 2.24). Larger values of λ move the initial search direction towards the direction of the negative gradient while smaller values result in a search direction nearer the linearization direction of equation (2.20). In this implementation, λ was initialized as 0.01.

3. Blocked Orthogonalization (BG)

In applying the blocked orthogonalization algorithm (BG), it was necessary to select the number of parameters j which were to be grouped in each block. In the experimental results recorded, the number of parameters chosen for each block was based on the natural grouping of parameters within the model being fitted. While the problems tested required groups of equal size, application of the algorithm to problems with varying block sizes would require only slight programming modifications.

The present implementation of the BG algorithm employed the nonlinear model considerations of (3.21). Selection of the one-dimensional search algorithm was made after monitoring the performances of several one-dimensional search techniques on various problems. A modified version of the one-dimensional quadratic fit of Aoki [69] was chosen for use since it was easy to apply and was consistently quick and accurate in predicting step lengths. The flowchart for this algorithm is presented in Appendix A.

4. Grey's Algorithm (GOOP)

The version of Grey's method (GOOP) used was obtained by taking j = 1 in the blocked algorithm. Hence, comparisons between BG and GOOP should be quite accurate since essentially the same code was used for both algorithms.

It is of interest to note that some users of GOOP employ methods other than the one-dimensional search for determining step length. Broste [56] for example, used a two stage correction procedure instead of a search. This technique could also be applied to the BG algorithm.

5. BG and GOOP Accelerations

The blocked algorithm and Grey's method were tested using Cornwell's acceleration technique [11,68]. To demonstrate the degree of accuracy required in the one-dimensional search used in this acceleration, two searches were employed. The BG-LAT and GOOP-LAT algorithms were implementations of the BG and GOOP algorithms in conjunction with the one-dimensional search LAT described by Cornwell. Appendix C contains a detailed flow chart of this algorithm. The BG-QF and GOOP-QF algorithms were implementations of BG and GOOP in which the quadratic fit routine was also employed in the acceleration step. The LAT routine represented a coarser search while the quadratic fit was a finer search.

6. Gauss-Hartley (GH)

Taking j = m in the blocked algorithm gives the implementation of the Gauss-Hartley algorithm used here. Hence, the BG, GOOP, and GH algorithms used essentially the same code.