

OPTIMIZATION METHODS
WITH APPLICATIONS TO
CHEMICAL ENGINEERING DESIGN

A thesis submitted for the
Ph.D. degree
of the University of London

by

Bruce Albert Murtagh

Department of Chemical Engineering
and Chemical Technology,
Imperial College of Science and Technology,
London, S.W.7.

August, 1969.

ABSTRACT

This thesis reports on the development and application of gradient methods of minimization that have quadratic termination properties. These properties extend to the case of minimization subject to linear constraints on the variables. Modifications have also been made to deal with nonlinear constraints.

The thesis is divided into three sections :

In Part 1, relevant previous work in the field of Nonlinear Programming is reviewed.

In Part 2, the methods of minimization are presented and their convergence properties established. Numerical examples are presented and discussed in relation to other methods.

In Part 3, the methods are applied to a specific problem in chemical engineering : that of an optimal design of a multicomponent distillation column.

ACKNOWLEDGEMENTS

The author is very pleased to acknowledge the untiring efforts and encouragement of his supervisor, Professor R.W.H. Sargent, who had a direct hand in every aspect of this work.

Thanks are also due to his colleagues in the computing section for their very real help, especially Martin J. Leigh, for providing much insight into the properties and problems associated with overall convergence.

The author would also like to record his thanks to the Commonwealth Scholarship Committee of the Association of Commonwealth Universities for their generous financial support.

CONTENTS

Page

PART 1 SURVEY OF RELEVANT NONLINEAR PROGRAMMING LITERATURE

CHAPTER 1 HILL-CLIMBING METHODS OF OPTIMIZATION

1.1 Unconstrained Minimization

1.1.1 Introduction	1
1.1.2 Classical First and Second-Order Methods	2
1.1.3 The Conjugate-Gradient Method	3
1.1.4 The Fletcher-Powell Method	4
1.1.5 Minimization Along a Line	5

1.2 Direct Methods of Handling Constraints

1.2.1 Introduction	7
1.2.2 Gradient-Projection Methods	7
1.2.3 Linear Approximation Techniques	10

1.3 Penalty Function Methods 12

CHAPTER 2 KUHN-TUCKER THEORY AND THE SADDLE-POINT PROBLEM

2.1 The Kuhn-Tucker Conditions for a Constrained Minimum

2.1.1 Lagrange Multipliers - Equality Constraints	14
2.1.2 Extension to Inequality Constraints - The Kuhn-Tucker Theorem	15
2.1.3 The Saddle-Point Problem	16

2.2 Solution of the Saddle-Point Problem

2.2.1 Direct Search for a Saddle Point	18
2.2.2 Transformation of Functions	19
2.2.3 Quadratic Approximation Programming	19

PART 2 MINIMIZATION METHODS WITH QUADRATIC CONVERGENCE

CHAPTER 3 MINIMIZATION WITHOUT CONSTRAINTS

3.1 Basis of the Methods	21
3.2 Construction of the Matrices, S_k	
3.2.1 Method A	23
3.2.2 Method B	25
3.3 Choice of Steps, p_k	28

CONTENTS

(continued)

	<u>Page</u>
<u>CHAPTER 3 (continued)</u>	
3.4 Procedure for Method A	31
3.5 Procedure for Method B	
3.5.1 Positive-Definiteness	34
3.5.2 Search Algorithms	37
3.6 Numerical Examples	41
3.7 Numerical Evaluation of Derivatives	46
 <u>CHAPTER 4 MINIMIZATION WITH CONSTRAINTS</u>	
4.1 The Projection Operator	49
4.2 Updating Required for S_{k-1}	53
4.3 Adding and Dropping Constraints	55
4.4 Kuhn-Tucker Theory and the Projection Operator	58
4.5 Constrained Step Strategy for Linear Constraints	
4.5.1 Algorithm	64
4.5.2 Linear Dependence of Hyperplanes	67
4.5.3 Degeneracy	68
4.5.4 Rounding Errors, x-correction	70
4.5.5 Initial Feasible Point and Starting Procedure	71
4.5.6 Numerical Examples	72
4.6 A Strategy for Nonlinear Constraints	
4.6.1 Introduction	75
4.6.2 Inequality Constraints	76
4.6.3 Penalty Terms	78
4.6.4 Projection for Feasibility	80
4.6.5 Algorithm	81
4.6.6 Numerical Examples	84

CONTENTS

(continued)

Page

CHAPTER 5 CONVERGENCE THEOREMS

5.1 Finite Convergence for Quadratic Functions

5.1.1 Minimization without Constraints 86

5.1.2 Minimization with Linear Constraints 89

5.2 Overall Convergence

5.2.1 Minimization without Constraints 92

5.2.2 Minimization with Linear Constraints 95

PART 3 OPTIMIZATION IN CHEMICAL ENGINEERING DESIGN

CHAPTER 6 PROGRAMME FOR THE CALCULATION OF A MULTICOMPONENT PLATE DISTILLATION COLUMN

6.1 Introduction 99

6.2 The Describing Equations 100

6.3 Solution of the Band Matrix 105

6.4 Convergence Techniques 106

CHAPTER 7 AN OPTIMIZATION PROBLEM IN DISTILLATION COLUMN DESIGN

7.1 Introduction 110

7.2 Product Specifications for a Single Plate 112

7.3 Product Specifications for Multiple Feedplates 114

7.4 Discussion of Results 116

GENERAL CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK 117

REFERENCES 119

APPENDIX 1 Unconstrained Minimization Results 122

APPENDIX 2 Linear Constraints 128

APPENDIX 3 Nonlinear Constraints 134

PART I

SURVEY OF RELEVANT
NONLINEAR PROGRAMMING LITERATURE

CHAPTER 1

HILL-CLIMBING METHODS OF OPTIMIZATION1.1 Unconstrained Minimization1.1.1 Introduction

In this section we consider the problem :

$$\text{minimize } f(\underline{x})$$

where the values of $\underline{x} = x_i$, $i = 1 \dots n$, the n independent variables are real numbers but otherwise have no constraints imposed on them.

The function is assumed to be smooth, in the sense that it possesses second-partial derivatives, which are continuous. Since such functions can be represented locally by a quadratic approximation, it is convenient to express $f(\underline{x})$ by a second-order Taylor's series expansion :

$$f(\underline{x}_k) = f(\underline{x}_{k-1}) + \underline{g}_{k-1}^T \underline{p}_k + \frac{1}{2} \underline{p}_k^T \underline{H}(\underline{x}_{k-1} + \theta \underline{p}_k) \underline{p}_k, \quad 0 \leq \theta \leq 1 \quad (1.1)$$

$$\text{where } \underline{p}_k = \underline{x}_k - \underline{x}_{k-1}$$

and $\underline{g}_k = \underline{g}(\underline{x}_{k-1})$, the gradient of the function at the point $\underline{x}_{k-1}$. $\underline{H}(\underline{x}_{k-1} + \theta \underline{p}_k)$ is the Hessian matrix of second partial derivatives evaluated at some point in the interval $(\underline{x}_k - \underline{x}_{k-1})$.

A necessary condition for the existence of an unconstrained minimum at the point $\hat{\underline{x}}$ is that the gradient there be zero :

$$\underline{g}(\hat{\underline{x}}) = \underline{0} \quad (1.2)$$

Equation (1.2) in itself only establishes $\hat{\underline{x}}$ as a stationary point. A sufficient condition that $\hat{\underline{x}}$ satisfying (1.2) is a minimum point (and not a maximum point or a point of inflection) is that the Hessian there be positive-definite :

$$\underline{p}^T \underline{H}(\hat{\underline{x}}) \underline{p} > 0 \quad \text{for all } \underline{p} \neq \underline{0} \quad (1.3)$$

1.1.2 Classical First and Second-Order Methods

The method of steepest descent, which dates back to Cauchy (1847), uses the negative gradient vector as the search direction :

$$\underline{p}_k = \underline{x}_k - \underline{x}_{k-1} = -\alpha_{k-1} \underline{g}_{k-1} \quad (1.4)$$

α_{k-1} is a scalar which determines the size of the step $\underline{p}_k$. It may be shown that this direction produces the greatest decrease in $f(\underline{x})$ in the locality of $\underline{x}_{k-1}$.

Despite its simplicity, the method can be quite powerful, especially if one is able to build up good empirical rules for the choice of α_{k-1} when applying it to a particular class of problems⁽¹⁾. Often a fixed step-length is used, and often the function is minimized along the direction of $-\underline{g}_{k-1}$. This would yield the minimum in one step if the contours of $f(\underline{x})$ were spherical, but since the contours are usually at best only ellipsoidal the path of the search follows a zig-zagging course towards the minimum. Hence the effectiveness of the method is most apparent at points far removed from the minimum, and its terminal convergence properties are poor.

The Newton-Raphson method is based on equation (1.1). If $f(\underline{x})$ was truly quadratic, so that the Hessian $\underline{H} = \underline{H}(\underline{x})$ was the same for all points $\underline{x}$, and hence positive-definite at all points, the minimum point could be reached in one step :

$$\underline{g}(\underline{x}) = \underline{g}_{k-1} + \underline{H}(\underline{x} - \underline{x}_{k-1}) \quad (1.5)$$

hence, using (1.2)

$$\hat{\underline{x}} = \underline{x}_k - \underline{H}^{-1} \underline{g}_{k-1} \quad (1.6)$$

For more general functions, (1.6) is used to define a direction of search :

$$\underline{p}_k = \underline{x}_k - \underline{x}_{k-1} = -\alpha_{k-1} \underline{H}_{k-1}^{-1} \underline{g}_{k-1} \quad (1.7)$$

It is apparent that the terminal convergence properties of this method are very good, since a quadratic is a valid local approximation of $f(\underline{x})$ about the minimum. On the other hand, the method is unlikely to be as efficient computationally as steepest descent at points far removed from the minimum. The first reason is that forming and inverting the Hessian matrix is a long process. Also, the Hessian at points far removed from the minimum is unlikely to be the same as that at the minimum, so this operation must be repeated, possibly at every step. Further, the Hessian thus calculated may not be positive-definite, in which case the method may not converge.

1.1.3 The Conjugate-Gradient Method

Because most functions can be represented locally by a quadratic, emphasis has been placed on methods which will give the minimum point of a quadratic function in a finite number of steps. In particular, it is apparent that one may use first-order information to achieve second-order convergence in $n+1$ steps.

The conjugate-gradient method was originally devised by Hestenes and Steifel⁽²⁾ as a method of solving a set of linear equations, and discussed subsequently by Fletcher and Reeves⁽³⁾ as a minimization method.

The search direction, $\underline{p}_k$, is not the negative gradient direction, but a linear combination of that and the previous direction :

$$\underline{p}_k = -\alpha_{k-1} (\underline{g}_{k-1} + \beta_{k-1} \underline{p}_{k-1}) \quad (1.8)$$

$$\text{where } \beta_{k-1} = \frac{\underline{g}_{k-1}^T \underline{g}_{k-1}}{\underline{g}_{k-2}^T \underline{g}_{k-2}}, \quad \beta_0 = 0 \quad (1.9)$$

It is essential that α_{k-1} be chosen to minimize $f(\underline{x})$ along the direction $\underline{p}_k$, so that $\underline{p}_k^T \underline{g}_k = 0$. It may be shown that, for quadratic functions, and if the initial search direction is the direction of steepest descent, the subsequent search directions thus chosen are mutually conjugate with

respect to the Hessian :

$$\mathbf{p}_k^T \mathbf{H} \mathbf{p}_j = 0 \quad j = 1 \dots k-1 \quad (1.10)$$

Hence the gradients at successive points are mutually orthogonal, and orthogonal to all previous steps :

$$\mathbf{g}_k^T \mathbf{g}_j = 0 \quad j = 1 \dots k-1 \quad (1.11a)$$

$$\mathbf{g}_k^T \mathbf{p}_j = 0 \quad j = 1 \dots k \quad (1.11b)$$

Thus, the minimum is achieved in n steps for a quadratic, since $\mathbf{g}_n$ is orthogonal to n linearly-independent vectors so must be zero.

The method is very appealing, for it achieves quadratic convergence without the need to store any approximation to the Hessian inverse. When it is applied to general functions, the need for accurate one-dimensional searches detracts from its usefulness. Also, since quadratic convergence is achieved only if the first direction is that of steepest descent, the process must be started afresh with this direction every (n) steps.

1.1.4 The Fletcher-Powell Method

This is another quadratically convergent minimization method, but differs from the conjugate-gradient method in that it uses the first order information to update an approximation to the inverse Hessian matrix. The current approximation, $\mathbf{S}_{k-1}$, is used to define the search direction :

$$\mathbf{p}_k = -\alpha_{k-1} \mathbf{S}_{k-1} \mathbf{g}_{k-1} \quad (1.12)$$

The matrix is updated using the symmetric correction :

$$\mathbf{S}_k = \mathbf{S}_{k-1} + \frac{\mathbf{p}_k \mathbf{p}_k^T}{\mathbf{p}_k^T \mathbf{g}_k} - \frac{\mathbf{S}_{k-1} \mathbf{g}_k \mathbf{g}_k^T \mathbf{S}_{k-1}}{\mathbf{g}_k^T \mathbf{S}_{k-1} \mathbf{g}_k} \quad (1.13)$$

$$\text{where } \underline{g}_k = \underline{g}_k - \underline{g}_{k-1} \quad (1.14)$$

In order to achieve quadratic convergence it is again essential to obtain a stationary minimum along the search direction, so that $\underline{p}_k^T \underline{g}_k = 0$. We then have $\underline{S}_n = \underline{H}^{-1}$ for quadratics, but the method generates conjugate directions and (1.11b) also holds, so that convergence is achieved in n steps. If the initial matrix, $\underline{S}_0$, is chosen as the identity matrix, the first direction is that of steepest descent, and in this case the Fletcher-Powell and conjugate-gradient methods follow identical paths on quadratics⁽⁴⁾.

The matrix-updating scheme given by (1.13) was devised by Davidon⁽⁵⁾, but the standard reference is to the paper by Fletcher and Powell⁽⁶⁾ who established the quadratically convergent nature of the method and presented a refined algorithm for its implementation.

Despite the need to store the matrix, $\underline{S}_k$, it has become the most well-known gradient method for unconstrained minimization. It is more powerful than the conjugate-gradient method when applied to general functions. Whether or not accurate one-dimensional searches are needed when the method is applied to general functions is a point of some contention in the literature, but in any case for most functions the process remains stable with $\underline{S}_k$ finite and positive-definite. This point is discussed further in Chapter 3.

1.1.5 Minimization Along a Line

There are two methods of minimizing along a line which are commonly used with gradient methods of optimization, or when the objective function is known to be reasonably smooth.

The first method fits a quadratic function along the line by using three data points : either the function value and gradient at the base point plus one other function value, or three function values. Rather than fit a higher order polynomial with function values at subsequent points, it is usual to fit a new quadratic function rejecting the worst of the previous points.

The second method fits a cubic function along the line by using the function value and gradient at two points. Again, the worst point is rejected and successive cubic fits yield minima which converge to the minimum of the objective function. This method is much more powerful than the quadratic fit, and is usually worth the computational expense of calculating gradients.

Both methods require precautions that the stationary point predicted by the quadratic or cubic approximation is a minimum and not a maximum.

1.2 Direct Methods of Handling Constraints

1.2.1 Introduction

The constrained minimization problem may be presented in the form :

$$\text{minimize } f^0(\underline{x}) \quad (1.15)$$

$$\text{subject to: } f^j(\underline{x}) \leq 0 \quad j = 1 \text{---} m^* \quad (1.16)$$

If any constraint is linear, it may be written as :

$$(\underline{g}^j)^T \underline{x} - b_j \leq 0 \quad (1.17)$$

Some constraints may hold as strict equalities, but in any case (1.16) describes a feasible region, within which $f^0(\underline{x})$ must be minimized. In this section we shall consider methods which take direct account of the constraints, insofar as they use them to restrict the path of the search.

1.2.2 Gradient-Projection Methods

The gradient projection method, first presented by Rosen in 1960⁽⁷⁾, was the first efficient method for handling linear constraints in non-linear optimization problems. Rosen's method is based on the steepest descent method of minimization.

Suppose that the current point lies on the boundary of the feasible region. For linear constraints, this boundary is formed by a set of hyperplanes given by the equation :

$$\underline{G}_m^T \underline{x} - \underline{b} = \underline{0} \quad (1.18)$$

i.e.

$$(\underline{g}^j)^T \underline{x} - b_j = 0 \quad j = 1 \text{---} m$$

A search direction which retains the steepest descent property as well

as feasibility is the orthogonal projection of the negative gradient onto the boundary :

$$\underline{p}_k = - \underline{P}_m \underline{g}_{k-1}^o \quad (1.19)$$

$$\text{where } \underline{P}_m = \underline{I} - \underline{G}_m (\underline{G}_m^T \underline{G}_m)^{-1} \underline{G}_m^T \quad (1.20)$$

$\underline{P}_m$ is an orthogonal projection operator; it leaves unchanged any vector in the boundary (i.e. the subspace defined by the intersection of constraining hyperplanes (1.18)), and it annihilates any vector orthogonal to this subspace. Thus, the search direction taken is that component of the negative gradient direction which lies in the boundary subspace.

A constraint is added to the basis (1.18) if its boundary is encountered during a search. Rosen has also devised a criterion for dropping a constraint (say j) :

$$\text{when } \left\| \underline{P}_m \underline{g}_k^o \right\| < \frac{1}{2} \left[(\underline{G}_m^T \underline{G}_m)^{-1} \underline{G}_m^T \underline{g}_k^o \right]_j / \left| b_{jj} \right|^{\frac{1}{2}} \quad (1.21)$$

where b_{jj} is the j^{th} diagonal element of $(\underline{G}_m^T \underline{G}_m)^{-1}$

$$\text{then } \left\| \underline{P}_{m-1} \underline{g}_k^o \right\| > \left\| \underline{P}_m \underline{g}_k^o \right\| \quad (1.22)$$

The procedure terminates where $\left\| \underline{P}_m \underline{g}_k^o \right\| = 0$ and $(\underline{G}_m^T \underline{G}_m)^{-1} \underline{G}_m^T \underline{g}_k^o \leq 0$ (within some error tolerance), so there is no constraint to drop. In this case Rosen furnishes theorems to show that a constrained overall minimum has been reached.

Simple recursion formulae may be used when adding and dropping a constraint from the basis. These and other aspects of this method are discussed further in Chapter 4.

Goldfarb and Lapidus⁽⁸⁾ have combined the gradient-projection technique with the method of Fletcher and Powell to produce an algorithm

which has quadratic convergence properties. Since the Fletcher-Powell matrix-updating scheme is valid only for steps p_k defined by $- \alpha_{k-1} \underline{S}_{k-1} \underline{E}_{k-1}^0$, Goldfarb and Lapidus incorporated projection properties into the matrix $\underline{S}_{k-1}$. Their matrix is the orthogonal projection of the current inverse Hessian approximation onto the subspace of active constraints. Thus for steps within this subspace, $\underline{S}_{k-1}$ is updated using the Davidon-Fletcher-Powell recursion (1.13), and when a constraint is added to or dropped from the basis, recursion formulae analogous to Rosen's are used. This procedure ensures that steps are in the subspace of active constraints, and whilst steps are confined to any one particular subspace they possess the quadratic convergence properties of the Fletcher-Powell method.

Besides the disadvantage of having to incorporate projection properties into the matrix $\underline{S}_{k-1}$ and thus having to discard accumulated information orthogonal to the current subspace, the method suffers from a more serious drawback. The inherent disadvantage of the Fletcher-Powell method in the presence of constraints is the need to go to a stationary minimum along each search direction, so that $p_k^T \underline{E}_k = 0$. Constraints will often prevent this occurring, and a new sequence of conjugate directions must be started each time the subspace of active constraints changes.

Davies⁽⁹⁾ has recently extended this method to deal with nonlinear constraints. The curvature of nonlinear constraints gives rise to serious problems, since one does not usually know 'a priori' the behaviour of the constraint function along the search direction. Davies' method involves a linearization of the nonlinear constraints about the current point, with the subsequent step taking due regard of these linearizations. This is followed by orthogonal projection back into

the feasible region if the resulting point violates any of the nonlinear constraints. The step-size is adjusted during the course of the search to take account of the curvature of the constraints.

1.2.3 Linear Approximation Techniques

Since nonlinear constraints may be conveniently handled by successive linearizations, a class of methods has been based on the extension of this idea to include successive linearization of the objective function. This gives rise to a sequence of linear programming problems, for which powerful methods of solution are well-established⁽¹⁰⁾.

The problem (1.15), (1.16) is thus reformulated as :

$$\text{minimize } f^0(\underline{x}_{k-1}) + (\underline{g}_{k-1}^0)^T(\underline{x} - \underline{x}_{k-1}) \quad (1.23)$$

$$\begin{aligned} \text{subject to } f^j(\underline{x}_{k-1}) + (\underline{g}_{k-1}^j)^T(\underline{x} - \underline{x}_{k-1}) &\leq 0 \\ j &= 1 \text{---} m^* \end{aligned} \quad (1.24)$$

The solution point, $\underline{x}_k$, of this problem is taken as the next base point, about which the objective and constraint functions are re-linearized.

The "Method of Approximation Programming", due to Griffiths and Stewart⁽¹¹⁾, uses this formulation; however, they also add a set of limitations on the step size to the set of constraints (1.24):

$$\underline{L} \leq \underline{x} - \underline{x}_{k-1} \leq \underline{U} \quad (1.24a)$$

Considerable success has been reported with this method, particularly in applications where the objective and constraint functions are not highly nonlinear.

Kelley's "Cutting Plane" method⁽¹²⁾ relinearizes only one constraint at a time and specifically considers a linear objective function. As he

points out, there is no loss of generality in this, since a nonlinear objective function may be replaced by a nonlinear constraint with a slack variable :

$$\text{minimize } x_{n+1} \quad , \quad \text{subject to } f^0(x_1 \dots x_n) - x_{n+1} = 0.$$

1.3 Penalty Function Methods

Since powerful methods of unconstrained minimization exist, it is an attractive proposition to transform a constrained problem into a sequence of unconstrained problems. This is especially so for highly nonlinear constraints, since direct methods of handling these will ultimately give rise to very small search steps.

The idea of adding a penalty term when a constraint is violated goes back to Courant in 1944⁽¹³⁾, and since then numerous penalty function formulations have appeared in the literature. They all modify the objective function so that it takes on very high values at or near a constraint boundary in order to force the search into the feasible region.

A typical formulation which allows constraint violations is as follows :

$$f(\underline{x}, K) = f^0(\underline{x}) + K \sum_{j=1}^{m^*} \delta(f^j(\underline{x})) \cdot (f^j(\underline{x}))^2 \quad (1.25)$$

$$\text{where } \delta(y) = \begin{cases} 0 & y \leq 0 \\ 1 & y > 0 \end{cases}$$

This is solved as an unconstrained problem for a sequence of increasing K .

Another formulation, which has been extensively developed by Fiacco and McCormick⁽¹⁴⁾, does not allow any constraint violations :

$$f(\underline{x}, r) = f^0(\underline{x}) - r \sum_{j=1}^{m^*} (1/f^j(\underline{x})) \quad (1.26)$$

The right-hand term goes to infinity as the constraint $f^j(\underline{x})$ approaches zero. This is solved as an unconstrained problem for a sequence of decreasing r . As r decreases, each solution gets closer to the boundary of the active constraints.

The unconstrained minimization problems arising from penalty function formulations are characterized by narrow curving valleys or ridges. Nevertheless, such formulations have become established as a most effective means of solving nonlinearly constrained problems.

CHAPTER 2

KUHN-TUCKER THEORY AND THE SADDLE-POINT PROBLEM2.1 The Kuhn-Tucker Conditions for a Constrained Minimum2.1.1 Lagrange Multipliers-Equality Constraints

Consider the problem :

$$\begin{array}{ll} \text{minimize} & f^0(\underline{x}) \\ \text{subject to} & f^j(\underline{x}) = 0 \quad j = 1 \text{ --- } m \end{array} \quad \left. \vphantom{\begin{array}{l} \text{minimize} \\ \text{subject to} \end{array}} \right\} \quad (2.1)$$

The classical approach to this problem is to adjoin the equality constraints to the objective function by means of Lagrange multipliers. We then have the unconstrained problem of finding a stationary point of the Lagrangian function :

$$L(\underline{x}, \underline{\lambda}) = f^0(\underline{x}) - \sum_{j=1}^m \lambda_j f^j(\underline{x}) \quad (2.2)$$

Under certain regularity assumptions, these two problems, (2.1) and (2.2), are equivalent⁽¹⁵⁾. At a stationary point, all the first partial derivatives of the function are zero.

$$\frac{\partial L(\underline{x}, \underline{\lambda})}{\partial x_i} = \frac{\partial f^0(\underline{x})}{\partial x_i} - \sum_{j=1}^m \lambda_j \frac{\partial f^j(\underline{x})}{\partial x_i} = 0 \quad i = 1 \text{ --- } n$$

$$\frac{\partial L(\underline{x}, \underline{\lambda})}{\partial \lambda_j} = - f^j(\underline{x}) = 0 \quad j = 1 \text{ --- } m$$

which is a set of $n+m$ equations for the $n+m$ unknowns (x_i, λ_j)
 $i = 1 \text{ --- } n, j = 1 \text{ --- } m.$

2.1.2 Extension to Inequality Constraints - The Kuhn-Tucker Theorem

Kuhn and Tucker⁽¹⁶⁾ have extended the use of Lagrange multipliers to inequality constrained minimization problems, and have put forward a set of necessary and sufficient conditions for the existence of a solution.

Theorem : If $\bar{\underline{x}}$ is a solution to the problem

$$\begin{aligned} & \text{minimize } f^0(\underline{x}) \\ & \text{subject to } f^j(\underline{x}) \leq 0 \quad j = 1 \text{ --- } m^* \end{aligned} \quad (2.3)$$

(Necessary Conditions) then there exists a vector $\bar{\underline{\lambda}}$ such that :

$$(i) \quad \frac{\partial L(\bar{\underline{x}}, \bar{\underline{\lambda}})}{\partial x_i} = \frac{\partial f^0(\bar{\underline{x}})}{\partial x_i} - \sum_{j=1}^{m^*} \bar{\lambda}_j \frac{\partial f^j(\bar{\underline{x}})}{\partial x_i} = 0 \quad i = 1 \text{ --- } n \quad (2.4)$$

$$(ii) \quad \bar{\lambda}_j \leq 0 \quad j = 1 \text{ --- } m^* \quad (2.5)$$

$$(iii) \quad \frac{\partial L(\bar{\underline{x}}, \bar{\underline{\lambda}})}{\partial \lambda_j} = -f^j(\bar{\underline{x}}) \geq 0 \quad j = 1 \text{ --- } m^* \quad (2.6)$$

$$(iv) \quad \bar{\underline{\lambda}}^T \frac{\partial L(\bar{\underline{x}}, \bar{\underline{\lambda}})}{\partial \underline{\lambda}} = \sum_{j=1}^{m^*} \lambda_j f^j(\bar{\underline{x}}) = 0 \quad (2.7)$$

(Sufficient Conditions) If, also, the following condition holds, then these conditions are sufficient.

$$(v) \quad L(\underline{x}, \bar{\underline{\lambda}}) \geq L(\bar{\underline{x}}, \bar{\underline{\lambda}}) + \frac{\partial L(\bar{\underline{x}}, \bar{\underline{\lambda}})}{\partial \underline{x}} (\underline{x} - \bar{\underline{x}}) \quad \text{for all } \underline{x} \in E^n \quad (2.8)$$

Note that (v) is satisfied if the Lagrangian is convex in $\underline{x}$. That this condition provides sufficiency is easily shown :

(Please see over.....)

from (ii) $\bar{\lambda}_j \leq 0 \quad j = 1 \dots m^*$

Note also $f^j(\underline{x}) \leq 0$ for all feasible $\underline{x}$.

Hence $\sum_{j=1}^{m^*} \bar{\lambda}_j f^j(\underline{x}) \geq 0$ for all feasible $\underline{x}$, and, using

(i) and (iv), (v) implies that

$$f^0(\underline{x}) - \sum_{j=1}^{m^*} \bar{\lambda}_j f^j(\underline{x}) \geq f^0(\bar{x})$$

i.e. $f^0(\underline{x}) \geq f^0(\bar{x})$ for all feasible $\underline{x}$.

Note that, using (ii) and (iii), (iv) implies that either $\bar{\lambda}_j = 0$ or $f^j(\bar{x}) = 0$. This simply means that constraints not active at $(\bar{x})$ are ignored.

The above conditions are subject to a regularity assumption when nonlinear constraints are present, namely that there is at least one feasible point for which all the nonlinear constraints are satisfied as strict inequalities⁽¹⁷⁾.

2.2.3 The Saddle-Point Problem

The necessary conditions for optimality given by the Kuhn-Tucker Theorem are also necessary conditions for the existence of a saddle-point of the Lagrangian function at $(\bar{x}, \bar{\lambda})$.

$$\begin{aligned} \text{i.e.} \quad L(\bar{x}, \underline{\lambda}) &\leq L(\bar{x}, \bar{\lambda}) \leq L(\underline{x}, \bar{\lambda}) & \underline{\lambda} &\leq \underline{0} \\ & & \underline{x} &\in E^n \end{aligned} \quad (2.9)$$

The sufficiency condition, that the Lagrangian be convex in $\underline{x}$, is also a sufficient condition for the existence of a saddle-point, since the Lagrangian is linear and hence concave, in $\underline{\lambda}$:

$$L(\bar{x}, \underline{\lambda}) = L(\bar{x}, \bar{\lambda}) + \frac{\partial L(\bar{x}, \bar{\lambda})}{\partial \underline{\lambda}} (\underline{\lambda} - \bar{\lambda})$$

using conditions (iii) and (iv) :

$$L(\underline{\bar{x}}, \underline{\lambda}) \leq L(\underline{\bar{x}}, \underline{\bar{\lambda}}) \quad \text{for all } \underline{\lambda} \leq \underline{0}.$$

The right-hand inequality of (2.9), follows immediately from (2.8) since

$$\frac{\partial L}{\partial \underline{x}}(\underline{\bar{x}}, \underline{\bar{\lambda}}) = \underline{0}.$$

Under these conditions, the solution of the original constrained minimization problem, (2.3), may be found by the solution of the saddle-point problem :

$$\min_{\underline{x} \in E^n} \max_{\underline{\lambda} \leq 0} L(\underline{x}, \underline{\lambda}) \tag{2.10}$$

Note that this minimization is over the complete space of $\underline{x}$, not just feasible $\underline{x}$. But from (iii) it follows that the solution point $(\underline{\bar{x}}, \underline{\bar{\lambda}})$ is feasible.

2.2 Solution of the Saddle-Point Problem

2.2.1 Direct Search for a Saddle-Point

Arrow, Hurwicz, and Uzawa⁽¹⁸⁾ have proposed a differential gradient procedure, which takes the form of two simultaneous differential equations :

$$\frac{d\underline{x}}{dt} = \frac{\partial f^0(\underline{x})}{\partial \underline{x}} - \sum_{j=1}^{m^*} \lambda_j \frac{\partial f^j(\underline{x})}{\partial \underline{x}}$$

$$\frac{d\lambda_j}{dt} = \begin{cases} f^j(\underline{x}) & \text{if } \lambda_j > 0 \text{ or } f^j(\underline{x}) > 0 \\ 0 & \text{otherwise} \end{cases}$$

Note that this constitutes a full search in the $E^n \times E^m$ space of $\underline{x}$ and $\underline{\lambda}$. It is only possible to prove that the process converges to the saddle point for a strictly convex objective function and strictly convex constraints. This places a severe limitation on the method.

King⁽¹⁹⁾ has proposed a Liapunov function :

$$\text{minimize } V(\underline{x}, \underline{\lambda}) = \left[\frac{\partial f^0(\underline{x})}{\partial \underline{x}} - \sum_{j=1}^{m^*} \lambda_j \frac{\partial f^j(\underline{x})}{\partial \underline{x}} \right]^T \left[\frac{\partial f^0}{\partial \underline{x}} - \sum_{j=1}^{m^*} \lambda_j \frac{\partial f^j}{\partial \underline{x}} \right] + \underline{f}^{**}(\underline{x})^T \underline{f}^{**}(\underline{x})$$

where $\underline{f}^{**}(\underline{x})$ is the set of strict equality constraints. This function is positive-definite everywhere except at a number of isolated points, which must be investigated for convergence. Note that it needs calculation of second-derivatives to minimize $V(\underline{x}, \underline{\lambda})$ by gradient methods.

2.2.2 Transformation of Functions

Another approach to the solution of the saddle-point problem is to transform the search in the spaces of $\underline{x}$ and $\underline{\lambda}$ into two separate sub-problems.

$$\text{Problem 1} \quad \underset{\underline{x} \in E^n}{\text{minimize}} \quad L(\underline{x}, \underline{\lambda}) = f^0(\underline{x}) - \sum_{j=1}^{m^*} \lambda_j f^j(\underline{x})$$

with $\underline{\lambda}$ fixed.

$$\text{Problem 2} \quad \underset{\underline{\lambda} \leq 0}{\text{maximize}} \quad \gamma(\underline{\lambda}) = \left\{ \min_{\underline{x} \in E^n} L(\underline{x}, \underline{\lambda}) \right\}$$

The function $\gamma(\underline{\lambda})$ is the envelope of the solutions of Problem 1 for various $\underline{\lambda}$.

This transformation of the original objective function $L(\underline{x}, \underline{\lambda})$ into the function $\gamma(\underline{\lambda})$ was mentioned in Courant and Hilbert⁽¹⁵⁾ for equality constraints. It has been discussed more recently by Takahashi⁽²⁰⁾, and generalized to include inequality constraints by Falk⁽²¹⁾.

Its effectiveness is limited by the effort in solving Problem 1 for each setting of $\underline{\lambda}$, and the need for convexity of the constraints and strict convexity of the objective function. However, some problems have a structure which is particularly amenable to this transformation; in particular Brosilow and Lasdon⁽²²⁾ have applied the technique to the decomposition of complex chemical plant.

2.2.3 Quadratic Approximation Programming

Using the Kuhn-Tucker conditions, Wilson⁽²³⁾ has developed an interesting extension of the linear approximation programming method discussed in section (1.2.3). The Taylor's series expansions of the

objective and constraint functions are truncated at the second-order term :

$$f^0(\underline{x}) = f^0(\underline{x}_k) + (\underline{g}_k^0)^T(\underline{x} - \underline{x}_k) + \frac{1}{2}(\underline{x} - \underline{x}_k)^T \underline{H}_k^0(\underline{x} - \underline{x}_k) + O\left\|\underline{x} - \underline{x}_k\right\|^2$$

$$f^j(\underline{x}) = f^j(\underline{x}_k) + (\underline{g}_k^j)^T(\underline{x} - \underline{x}_k) + \frac{1}{2}(\underline{x} - \underline{x}_k)^T \underline{H}_k^j(\underline{x} - \underline{x}_k) + O\left\|\underline{x} - \underline{x}_k\right\|^2$$

$$\leq 0 \qquad j = 1 \text{ --- } m^*$$

A sequence of quadratic programming problems is generated, using Lagrange multipliers to put the nonlinearities in the constraint functions into the objective function :

$$\begin{aligned} \text{minimize } & f^0(\underline{x}_k) + (\underline{g}_k^0)^T(\underline{x} - \underline{x}_k) + \frac{1}{2}(\underline{x} - \underline{x}_k)^T \left(\underline{H}_k^0 - \sum_{j=1}^{m^*} \lambda_j \underline{H}_k^j \right) (\underline{x} - \underline{x}_k) \\ \text{subject to } & f^j(\underline{x}_k) + (\underline{g}_k^j)^T(\underline{x} - \underline{x}_k) \leq 0 \qquad j = 1 \text{ --- } m^* \end{aligned}$$

The values of the Lagrange multipliers, λ_j , used are those at the solution of the previous problem in the sequence (and must be guessed for the first problem).

Note in particular that, as well as the constraints, the objective function is also truncated at second-order. A serious drawback to the method is the need to calculate $(m^* + 1)$ Hessian matrices of second partial-derivates for each member of the sequence of problems. However, it is remarked in the reference that it is not necessary to recalculate these at every step of the sequence. For large m^* , the storage of the matrices may in itself present a problem.

PART II

MINIMIZATION METHODS

WITH QUADRATIC CONVERGENCE

CHAPTER 3

MINIMIZATION WITHOUT CONSTRAINTS3.1 Basis of the Methods

The principle of solution is to assume a local quadratic approximation to the function and construct successive approximations to the inverse of the Hessian matrix, using these to estimate the position of the minimum at each step.

Thus we consider a function, $f(\underline{x})$, of n variables, with continuous second derivatives, and expand it in a Taylor series with a remainder term of second order :

$$f_k = f_{k-1} + \underline{g}_{k-1}^T \underline{p}_k + \frac{1}{2} \underline{p}_k^T \underline{H}(\underline{x}_{k-1} + \theta \underline{p}_k) \underline{p}_k \quad 0 \leq \theta \leq 1 \quad (3.1)$$

where $f_k \equiv f(\underline{x}_k)$, $\underline{g}_{k-1} \equiv \underline{g}(\underline{x}_{k-1})$ is the gradient of the function at point $\underline{x}_{k-1}$, $\underline{p}_k = \underline{x}_k - \underline{x}_{k-1}$, and $\underline{H}$ is the symmetric Hessian matrix.

The change in the gradient is then given by

$$\underline{g}_k = \underline{g}_k - \underline{g}_{k-1} = \underline{H} \cdot \underline{p}_k \quad (3.2)$$

Now if $\underline{x}_k = \hat{\underline{x}}$, the position of the minimum, we have $\underline{g}(\hat{\underline{x}}) = 0$, and provided that $\underline{H}$ is non-singular, equation (3.2) yields

$$\hat{\underline{x}} = \underline{x}_{k-1} - \underline{H}^{-1} \cdot \underline{g}_{k-1} \quad (3.3)$$

We are therefore interested in constructing successive approximations, $\underline{S}_k$, to the inverse of the Hessian matrix, and it seems reasonable to require that the approximation at any step should satisfy equation (3.2) for all steps so far made :

$$\underline{S}_k \underline{g}_j = \underline{p}_j \quad , \quad 1 \leq j \leq k \quad (3.4)$$

After n steps we can form the $n \times n$ matrices, $\underline{P}_n$ and $\underline{Q}_n$ with columns $\underline{p}_j$, $1 \leq j \leq n$, and $\underline{q}_j$, $1 \leq j \leq n$, respectively. Equation (3.4) then gives

$$\underline{S}_n \cdot \underline{Q}_n = \underline{P}_n \quad (3.5)$$

Now if the function $f(\underline{x})$ is adequately represented by a quadratic function over these n steps, the Hessian matrix $\underline{H}$ will not vary, and if further $\underline{H}$ and $\underline{Q}_n$ are non-singular, we have from (3.2) and (3.5) :

$$\underline{S}_n = \underline{P}_n \underline{Q}_n^{-1} = \underline{H}^{-1} \quad (3.6)$$

We can then obtain the position of the minimum from equation (3.3).

3.2 Construction of the matrices $\underline{S}_k$

A number of methods suggest themselves for the construction of $\underline{S}_k$ to satisfy equation (3.4) and here we consider two related methods.

3.2.1 Method A

The most obvious method is based on equation (3.5), written in the form

$$\underline{S}_k = \underline{P}_k \underline{R}_k \quad (3.7)$$

where $\underline{R}_k = \underline{Q}_k^{-1}$

We start with arbitrary matrices $\underline{P}_0$ and $\underline{Q}_0$, subject only to the condition that $\underline{Q}_0$ is non-singular, and at each step successively replace corresponding columns with the calculated vectors, $\underline{p}_k$ and $\underline{q}_k$:

$$\underline{P}_k = \underline{P}_{k-1} + (\underline{p}_k - \underline{P}_{k-1} \underline{e}_j) \cdot \underline{e}_j^T \quad (3.8)$$

$$\underline{Q}_k = \underline{Q}_{k-1} + (\underline{q}_k - \underline{Q}_{k-1} \underline{e}_j) \cdot \underline{e}_j^T \quad (3.9)$$

The corresponding inverse, $\underline{R}_k$, is readily computed by the recursion formula ⁽²⁴⁾:

$$\underline{R}_k = \underline{R}_{k-1} + \frac{(\underline{e}_j - \underline{R}_{k-1} \underline{q}_k) \cdot \underline{e}_j^T \underline{R}_{k-1}}{\underline{e}_j^T \underline{R}_{k-1} \underline{q}_k} \quad (3.10)$$

Note also that :

$$\det \left| \underline{Q}_k \right| = \det \left| \underline{Q}_{k-1} \right| \det \left| (\underline{I} + (\underline{R}_{k-1} \underline{q}_k - \underline{e}_j) \underline{e}_j^T) \right|$$

Expanding the right-hand determinant by diagonal elements ⁽²⁵⁾ we have :

$$\det \left| \underline{Q}_k \right| = \det \left| \underline{Q}_{k-1} \right| \cdot \underline{e}_j^T \underline{R}_{k-1} \underline{q}_k \quad (3.11)$$

Now if $\det \underline{Q}_{k-1} \neq 0$, $\underline{R}_{k-1}$ is non-singular and $\underline{R}_{k-1} \underline{q}_k$ cannot be null unless $\underline{q}_k$ itself is null. Otherwise there will be at least one non-zero element of the vector $\underline{R}_{k-1} \underline{q}_k$, so that we can choose $\underline{e}_j$ to ensure that $\underline{R}_k$ is non-singular. It would seem desirable to choose $\underline{e}_j$ so that $\underline{e}_j^T \underline{R}_{k-1} \underline{q}_k$ is the element of greatest absolute value, which makes $\det \underline{Q}_k$ as large as possible, but for general functions this may cause the same column to be replaced repeatedly. Since the quadratic is only a local approximation, it is equally important to retain the most recent information and a suitable compromise is to replace the oldest column for which $\underline{e}_j^T \underline{R}_{k-1} \underline{q}_k$ does not fall below a prescribed lower bound, chosen to ensure that a singular $\underline{Q}_k$ does not arise through rounding errors.

It is convenient to write equation (3.10) in the form :

$$\underline{R}_k = \underline{R}_{k-1} + (\underline{e}_j - \underline{R}_{k-1} \underline{q}_k) \cdot \underline{y}_k^T \quad (3.12)$$

$$\text{with } \underline{y}_k^T = \frac{\underline{e}_j^T \underline{R}_{k-1}}{\underline{e}_j^T \underline{R}_{k-1} \underline{q}_k} \quad (3.13)$$

Substitution of (3.8), (3.12) and (3.13) into equation (3.7) then yields a simple recursion formula for $\underline{S}_k$:

$$\underline{S}_k = \underline{S}_{k-1} + (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) \cdot \underline{y}_k^T \quad (3.14)$$

It is not therefore necessary to construct the $\underline{P}_k$ and $\underline{Q}_k$ explicitly : we merely start with arbitrary non-singular matrices $\underline{S}_0$ and $\underline{R}_0$ and up-date them using (3.12) - (3.14). If points are rejected for which $\underline{q}_k = 0$ and $\underline{e}_j$ is chosen as explained above, all successive $\underline{R}_k$ are non-singular, and the $\underline{S}_k$ satisfy equation (3.4) for all vectors $\underline{p}_j$ and $\underline{q}_j$ implicit in the current $\underline{P}_k$ and $\underline{Q}_k$ respectively.

3.2.2 Method B

Since $\underline{H}$ and $\underline{H}^{-1}$ are symmetric, it is of interest to investigate the possibility of constructing symmetric $\underline{S}_k$ to satisfy equation (3.4). Since any symmetric matrix can be expressed as a sum of symmetric simple products of type $\underline{z} \cdot \underline{z}^T$, we investigate the recursion formula :

$$\underline{S}_k = \underline{S}_{k-1} + \underline{z}_k \cdot \underline{z}_k^T / c_k \quad (3.15)$$

where $\underline{S}_{k-1}$ is symmetric and $\underline{z}_k$ is to be chosen to satisfy equation (3.4).

Setting $j = k$ in equation (3.4) and substituting (3.15) we have :

$$\underline{p}_k = \underline{S}_k \underline{q}_k = \underline{S}_{k-1} \underline{q}_k + \underline{z}_k \cdot \underline{z}_k^T \underline{q}_k / c_k$$

$$\left. \begin{array}{l} \text{Whence, if we choose } c_k = \underline{z}_k^T \underline{q}_k \\ \text{we obtain } \underline{z}_k = \underline{p}_k - \underline{S}_{k-1} \underline{q}_k \end{array} \right\} \quad (3.16)$$

From (3.15) and (3.16) :

$$\underline{S}_k = \underline{S}_{k-1} + \frac{(\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) \cdot (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k)^T}{\underline{q}_k^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k)} \quad (3.17)$$

We may note that this is also of the form of equation (3.14), where $\underline{y}_k$ is now given by :

$$\underline{y}_k = \frac{\underline{z}_k}{c_k} \quad (3.18)$$

It remains to demonstrate that $\underline{S}_k$, constructed according to equation (3.17), satisfies equation (3.4) for all preceding steps as well as the current step, and it turns out that this is only true if $f(\underline{x})$ may be represented by a quadratic function over the steps in question.

The proof is by induction.

We accordingly suppose that :

$$\underline{S}_{k-1} \underline{q}_j = \underline{p}_j \quad 1 \leq j < k \quad (3.19)$$

Then from (3.2) and (3.19) :

$$\underline{q}_k^T \underline{S}_{k-1} \underline{q}_j = \underline{q}_k^T \underline{p}_j = \underline{p}_k^T \underline{H} \underline{p}_j = \underline{p}_k^T \underline{q}_j \quad (3.20)$$

and from the definition of $\underline{z}_k$ in (3.16) :

$$\underline{z}_k^T \underline{q}_j = \underline{p}_k^T \underline{q}_j - \underline{q}_k^T \underline{S}_{k-1} \underline{q}_j = 0 \quad (3.21)$$

Hence, from (3.15) :

$$\underline{S}_k \underline{q}_j = \underline{S}_{k-1} \underline{q}_j = \underline{p}_j \quad 1 \leq j < k \quad \text{as required.}$$

Now $\underline{S}_k$ has been constructed so that $\underline{S}_k \underline{q}_k = \underline{p}_k$, and in particular $\underline{S}_1 \underline{q}_1 = \underline{p}_1$ for arbitrary $\underline{S}_0$. This completes the proof.

Essentially both methods are methods for solving the set of non-linear equations : $\underline{g}(\underline{x}) = 0$, for which $\underline{H}$ is the Jacobian matrix.

Method A makes no assumptions about the nature of $\underline{H}$, other than that it is non-singular, and it may therefore be used for solving general sets of non-linear equations.

In both methods the matrix $\underline{S}_k$ must be stored, and Method A further requires the storage of a second auxiliary matrix ($\underline{R}_k$). It will be seen in Chapter 4 that the flexibility introduced by the use of $\underline{R}_k$ in Method A can be used to advantage in dealing with constraints. Method B on the other hand has the advantage of requiring less storage space and less computation per step but cannot be used for solving sets of

equations with non-symmetric $\underline{H}$.

Broyden⁽²⁶⁾ has discussed a more general class of methods based on the recursion :

$$\underline{S}_k = \underline{S}_{k-1} + p_k \underline{y}_k^T - \underline{S}_{k-1} \underline{q}_k \cdot \underline{z}_k^T$$

This includes the Fletcher-Powell method as well as the two methods presented here, and Broyden mentions the generalized-secant method and Method B above but does not discuss computational performance. Davidon⁽²⁷⁾ has also independently arrived at the same matrix updating scheme as Method B. Some of the results in this Chapter have been presented in an earlier paper⁽²⁸⁾.

3.3 Choice of Steps, $\underline{p}_k$

It is logical to choose the step $\underline{p}_k$ according to equation (3.3), replacing $\underline{H}^{-1}$ by its current approximation, $\underline{S}_{k-1}$. However, if this is a poor approximation such a step may well lead to an increase in the function, and it is wiser to allow for a modification of the distance along the direction of the predicted minimum according to the formula :

$$\underline{p}_k = -\alpha_{k-1} \underline{S}_{k-1} \underline{E}_{k-1} \quad (3.22)$$

where α_{k-1} is a suitably chosen scalar.

If we ensure that the function $f(\underline{x})$ initially decreases as we move from $\underline{x}_{k-1}$ along the direction $\underline{p}_k$, we can always choose an α_{k-1} such that $f(\underline{x}_k) < f(\underline{x}_{k-1})$. We must therefore have $\underline{E}_{k-1}^T \underline{p}_k < 0$, and so long as $\underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} \neq 0$ we can always choose the sign of α_{k-1} so that this is so.

In chapter 5 we shall show that in order to achieve convergence we need to make these conditions slightly stronger, according to the relations :

$$\rho \left\| \underline{E}_k \right\| \leq \left\| \underline{S}_k \underline{E}_k \right\| \leq \sigma \left\| \underline{E}_k \right\| \quad (3.23)$$

$$\left| \underline{E}_k^T \underline{S}_k \underline{E}_k \right| \geq \xi \left\| \underline{E}_k \right\| \cdot \left\| \underline{S}_k \underline{E}_k \right\| \quad (3.24)$$

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) \geq \delta \cdot \alpha_k \cdot \underline{E}_k^T \underline{S}_k \underline{E}_k > 0 \quad (3.25)$$

$$\left| \alpha_k \right| \geq \gamma \quad (3.26)$$

where $\rho, \xi, \sigma, \delta, \gamma$ are fixed positive constants.

Condition (3.23) represents upper and lower bounds on the norm of the vector $\underline{S}_k \underline{g}_k$ in relation to the vector $\underline{g}_k$. As will be seen in Chapter 5, the lower bound is essential in order to ensure that the sequence of functions converge to where $\underline{g}_k = 0$, and the upper bound is needed to ensure that the steps $\underline{p}_k \rightarrow 0$ as $\underline{g}_k \rightarrow 0$.

Condition (3.24) limits the angle between the search direction and that of steepest descent, so that with α_k of the appropriate sign the function initially decreases along the search direction. This is associated with condition (3.25) which bounds the function decrease above a positive multiple of $\underline{g}_k^T \underline{S}_k \underline{g}_k$, so that together these two conditions ensure that the steps do not stop short of a stationary point. Condition (3.26) is of course essential; it postulates that the search steps do not become infinitely small.

It will be seen later that non-singularity of the approximating matrix $\underline{S}_k$ is a useful property to retain, and in any case this ensures that the left-hand side of (3.23) is satisfied. Consider the determinant of the updated matrix :

From (3.17) and (3.18) we have :

$$\det |\underline{S}_k| = \det |\underline{S}_{k-1}| \cdot \det \left\{ \underline{I} + (\underline{S}_{k-1}^{-1} \underline{p}_k - \underline{q}_k) \underline{y}_k^T \right\} \quad (3.27)$$

Expanding the right hand determinant by diagonal elements, and noting that $\underline{y}_k^T \underline{q}_k = 1$, we have :

$$\det |\underline{S}_k| = \det |\underline{S}_{k-1}| \cdot \underline{y}_k^T \underline{S}_{k-1}^{-1} \underline{p}_k \quad (3.28)$$

using (3.22)

$$\det |\underline{S}_k| = -\det |\underline{S}_{k-1}| \cdot \alpha_{k-1} \underline{y}_k^T \underline{g}_{k-1} \quad (3.29)$$

Hence, a necessary and sufficient condition for $\underline{S}_k$ to be non-singular (with $\underline{S}_{k-1}$ non-singular and $\alpha_{k-1} \neq 0$) is that

$$\underline{y}_k^T \underline{E}_{k-1} \neq 0 \quad (3.30)$$

Thus, if $\underline{S}_0$ is non-singular and we ensure that (3.26) and (3.30) are satisfied at each step, all successive $\underline{S}_k$ are non-singular.

All the methods break down if $\underline{q}_k = 0$, for $\underline{q}_k$ occurs in the denominator of $\underline{y}_k$ in each case. This means that $\underline{S}_{k-1}$ cannot be updated, and since $\underline{E}_k = \underline{E}_{k-1}$, the direction of search is also unchanged. In general of course there will only be a finite set of points for which $\underline{q}_k = 0$ so that a change of α_{k-1} will usually be sufficient to avoid the problem. However, it may arise that $\underline{g}(\underline{x})$ is constant for all points along the line of search and in this case the simplest course of action is to make a step in the steepest descent direction without updating $\underline{S}_{k-1}$ or the auxiliary matrices. It may be noted that this also constitutes a failing case for the Fletcher-Powell method.

With these precautions the procedures will generate a sequence of points $\underline{x}_k$ such that $f(\underline{x}_k) \leq f(\underline{x}_{k-1})$, where the equality occurs only if $\underline{E}_{k-1} = 0$, indicating that $\underline{x}_{k-1}$ is a stationary point. A local search about $\underline{x}_{k-1}$, for example along each of the coordinate directions, will then either confirm that this is a minimum or else provide a new point with a smaller function value, from which the procedure can be continued.

We must now examine how these conditions can be met in practice for the proposed methods.

3.4 Procedure for Method A

We consider first the choice of α_k to satisfy (3.24).

At step $(k-1)$, $\underline{g}_{k-1}^T \underline{S}_{k-1} \underline{g}_{k-1}$ satisfied (3.24), and since $g(\underline{x})$ is a continuous function of $\underline{x}$, there is a neighbourhood of $\underline{x}_{k-1}$ for which $|\underline{g}_k^T \underline{S}_{k-1} \underline{g}_k| / \|\underline{g}_k\| \cdot \|\underline{S}_{k-1} \underline{g}_k\|$ satisfies (3.24) and we can choose α_{k-1} to give an $\underline{x}_k$ in this neighbourhood. Now from equation (3.14) we have :

$$\underline{g}_k^T \underline{S}_k \underline{g}_k = \underline{g}_k^T \underline{S}_{k-1} \underline{g}_k + \underline{g}_k^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) \cdot \underline{y}_k^T \underline{g}_k \quad (3.31)$$

If the second term on the right hand side should be of such sign and magnitude as to prevent $\underline{g}_k^T \underline{S}_k \underline{g}_k$ from satisfying the condition, we can always make $\underline{y}_k^T \underline{g}_k = 0$, and hence guarantee that condition (3.24) is satisfied for a sufficiently small α_{k-1} .

From equations (3.13) and (3.29) we see that $\underline{y}_k^T \underline{g}_k = 0$ makes $\underline{y}_k^T \underline{g}_{k-1} = -1$, which also conveniently satisfies the condition for non-singularity of $\underline{S}_k$. Thus, when corrections according to equations (3.12) - (3.14) with the normal choice of $\underline{e}_j$ result in either $\underline{y}_k^T \underline{g}_{k-1} = 0$, or condition (3.24) not being satisfied, we can make further corrections according to the formulae :

$$\begin{aligned} \underline{R}'_k &= \underline{R}_k + (\underline{e}_m + \underline{R}_k \underline{g}_{k-1}) \underline{z}_k^T \\ \underline{S}'_k &= \underline{S}_{k-1} + (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) (\underline{y}'_k)^T \end{aligned} \quad (3.32)$$

$$\text{where } \underline{z}_k^T = \frac{-\underline{e}_m^T \underline{R}_k}{\underline{e}_m^T \underline{R}_k \underline{g}_{k-1}} \quad \text{and} \quad (\underline{y}'_k)^T = (\underline{e}_j + \underline{e}_m)^T \underline{R}'_k$$

Note that the result $\underline{y}_k^T \underline{g}_{k-1} = 0$ leads to a simpler correction :

$$\underline{S}'_k = \underline{S}_k + (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) \underline{z}_k^T$$

The vector $\underline{e}_m$ is chosen by the same rule as $\underline{e}_j$, but with the restriction that $m \neq j$. The combined correction factor $\underline{y}'_k$ which generates $\underline{S}'_k$ from $\underline{S}_{k-1}$ is then equivalent to $\underline{y}_k$ in (3.14) and is orthogonal to $\underline{g}_k$ as required.

This result may perhaps be more easily seen in relation to the matrix $\underline{Q}_k$. It was noted (c.f. equations 3.10 and 3.12) that we require $\underline{y}_k$ to be the solution of $\underline{Q}_k^T \underline{y}_k = \underline{e}_j$. To make $\underline{y}_k^T \underline{g}_{k-1} = -1$, we effectively put $-\underline{g}_{k-1}$ in column m of $\underline{Q}_k$ and solve $(\underline{Q}'_k)^T \underline{y}'_k = \underline{e}_j + \underline{e}_m$.

If in fact this condition is not necessary to satisfy (3.24), we may be able to obtain $\underline{y}_k^T \underline{g}_{k-1} \neq 0$ more simply by continuing the search for an $\underline{e}_j$ which yields this result as well as a suitably large value of $\underline{e}_j^T \underline{R}_{k-1} \underline{q}_k$ in the first correction.

It is now possible to give a complete statement of the procedure for step k given $\underline{x}_{k-1}$, $\underline{g}_{k-1}$, $\underline{R}_{k-1}$ and $\underline{S}_{k-1}$:

1. Choose the sign of α_{k-1} to make $\alpha_{k-1} \underline{g}_{k-1}^T \underline{S}_{k-1} \underline{g}_{k-1}$ positive
2. Starting with $|\alpha_{k-1}| = 1$, and computing $\underline{p}_k$ from (3.22), determine a point $\underline{x}_k$ with $0 \leq |\alpha_{k-1}| \leq 1$ to satisfy (3.25), using some search method for minimizing along a line or by successive halving if necessary.
3. Compute $\underline{g}_k = \underline{g}(\underline{x}_k)$, and if $\|\underline{g}_k\| \leq \epsilon$ proceed to the local search procedure for confirmation of a minimum. Otherwise go to step 4.
4. If $|\underline{g}_k^T \underline{S}_{k-1} \underline{g}_k| / \|\underline{g}_k\| \cdot \|\underline{S}_{k-1} \underline{g}_k\| < \delta$, return to step 2 and seek a smaller value of $|\alpha_{k-1}|$. (If $|\alpha_{k-1}|$ becomes less than δ register

a failure of this step for purposes of step 7.)

5. Check that $|\alpha_{k-1}| \geq \delta$ and $\|\underline{S}_{k-1} \underline{E}_{k-1}\| / \|\underline{E}_{k-1}\| < \gamma$; rescale α_{k-1} and $\underline{S}_{k-1}$ if necessary.
6. Compute $\underline{y}_k$ and $\underline{R}_k$ from equations (3.12) and (3.13) choosing the oldest column for which $|\underline{e}_j^T \underline{R}_{k-1} \underline{q}_k| \geq \delta$, and if possible $|\underline{y}_k^T \underline{E}_{k-1}| \geq \delta$. If it is not possible to satisfy this latter condition, correct $\underline{R}_k$ and $\underline{S}_k$ using equation (3.32).
7. Test that condition (3.24) is satisfied. If it is not, correct $\underline{R}_k$ and $\underline{S}_k$ using (3.32) if the test in step 4 was satisfied, otherwise reset $\underline{S}_k = \underline{I}$, $\underline{R}_k = \underline{I}$.

In this procedure ξ and δ are small positive constants chosen to ensure that no difficulties arise through rounding errors.

3.5 Procedure for Method B

3.5.1 Positive-Definiteness

For Method B we can show that, if $\underline{S}_0$ is chosen to be positive definite and we can always choose α_{k-1} so that $\underline{y}_k^T \underline{E}_{k-1} < 0$, then all $\underline{S}_k$ are positive definite. The proof is by induction, so we suppose that $\underline{S}_{k-1}$ is positive definite and $\underline{E}_{k-1} \neq 0$.

We note first that $\underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} > 0$ so that α_{k-1} must be positive to obtain $\underline{E}_{k-1}^T \underline{p}_k < 0$. Now consider the matrix $\underline{S}(t)$ defined by :

$$\underline{S}(t) = \underline{S}_{k-1} + t(\underline{p}_k - \underline{S}_{k-1} \underline{a}_k) \underline{y}_k^T \quad (3.33)$$

where t is a scalar in the interval $0 \leq t \leq 1$.

From (3.33) :

$$\det |\underline{S}(t)| = \det |\underline{S}_{k-1}| \cdot \{ (1-t) - t \underline{y}_k^T \underline{E}_{k-1} \alpha_{k-1} \} \quad (3.34)$$

Since $\det |\underline{S}_{k-1}| \neq 0$ and $\underline{y}_k^T \underline{E}_{k-1} < 0$ by hypothesis, and $\alpha_{k-1} > 0$, it follows that the right hand side of (3.34) cannot vanish, and hence that $\underline{S}(t)$ is non-singular, over the interval $0 \leq t \leq 1$. Now according to a theorem given by Caratheodory⁽²⁹⁾, a matrix $\underline{S}(t)$ is positive definite in each interval $t_1 \leq t \leq t_2$ for which $\underline{S}(t)$ is non-singular, provided that this interval contains a point t_0 for which $\underline{S}(t_0)$ is positive definite. The point $t = 0$ is such a point in the interval $0 \leq t \leq 1$, and $\underline{S}(t)$ is therefore positive definite over this interval. In particular, $\underline{S}_k = \underline{S}(1)$ is positive definite and this completes the proof.

To examine the conditions for $\underline{y}_k^T \underline{E}_{k-1} < 0$ we consider separately the numerator and denominator of $\underline{y}_k$. Substituting from (3.22) we have :

(Please see over ...)

$$\begin{aligned} \underline{E}_{k-1}^T \underline{z}_k &= \underline{E}_{k-1}^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) = (1 - \alpha_{k-1}) \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} - \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_k \\ &\dots (3.35) \end{aligned}$$

$$\begin{aligned} c_k &= \underline{q}_k^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) = (2 - \alpha_{k-1}) \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_k - (1 - \alpha_{k-1}) \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} - \\ &\quad - \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_k \dots (3.36) \end{aligned}$$

Using (3.35) to eliminate α_{k-1} from (3.36), together with (3.16) and (3.18), we obtain after some rearrangement :

$$\begin{aligned} \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} c_k &= \underline{z}_k^T \underline{E}_{k-1} \left[\underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_k - \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} \right] - \\ &\quad - \left[\underline{E}_k^T \underline{S}_{k-1} \underline{E}_k \cdot \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} - (\underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_k)^2 \right] (3.37) \end{aligned}$$

Now since $\underline{S}_{k-1}$ is positive definite the second term on the right hand is always positive, by the Schwarz inequality, unless $\underline{E}_k$ is parallel to $\underline{E}_{k-1}$, when it is zero.

It follows from (3.35) that if $\underline{z}_k^T \underline{E}_{k-1} > 0$, then $\underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_{k-1} > \underline{E}_{k-1}^T \underline{S}_{k-1} \underline{E}_k$ and the first term on the right hand side of (3.37) is negative.

Thus, if $\underline{z}_k^T \underline{E}_{k-1} > 0$, we must have $c_k < 0$ and the condition $\underline{y}_k^T \underline{E}_{k-1} < 0$ is satisfied. If this occurs for $\alpha_{k-1} \geq 1$ equation (3.35) shows that $\underline{p}_k^T \underline{E}_k > 0$ and hence a minimum must occur along the line between $\underline{x}_{k-1}$ and $\underline{x}_k$.

If $c_k > 0$ we must have $\underline{z}_k^T \underline{E}_{k-1} < 0$ and again the condition $\underline{y}_k^T \underline{E}_{k-1} < 0$ is satisfied. In this case (3.35) shows that if $0 < \alpha_k \leq 1$ we have $\underline{p}_k^T \underline{E}_k < 0$ and the function is decreasing at $\underline{x}_k$; unless the

function is quite badly behaved it will not have both a maximum and a minimum in this range of α_{k-1} so that this normally implies that $f(\underline{x}_k) < f(\underline{x}_{k-1})$.

It is unfortunately possible that $c_k \leq 0$ and $\underline{z}_k^T \underline{g}_{k-1} \leq 0$ together (although both can be simultaneously zero only if $\underline{g}_k$ is parallel to $\underline{g}_{k-1}$), so that the condition $\underline{y}_k^T \underline{g}_{k-1} < 0$ can fail. If this failure occurs at $\alpha_{k-1} = 1$, then we have $\underline{p}_k^T \underline{g}_k \leq 0$, with equality if $\underline{z}_k^T \underline{g}_{k-1} = 0$. In this latter case the stationary point will normally be a minimum, correctly predicted using $\underline{S}_{k-1}$ so it seems reasonable simply to set $\underline{S}_k = \underline{S}_{k-1}$; in any case we shall normally have obtained a function decrease, as explained above. An interesting result is that, for $\alpha_{k-1} = 1$, the next search direction, $\underline{S}_k \underline{g}_k$, is only sign-dependent on c_k .

In fact we are left with little theoretical guidance as to the best course of action when the method does fail. The most obvious course is to reset $\underline{S}_k$ to the identity matrix, and thus restart with the steepest descent direction. Some preliminary numerical experiments indicated that resetting $\underline{S}_k = \underline{S}_{k-1}$ often led to repeated subsequent failure; however, it is evident from (3.21) that it is possible to retain the information in $\underline{S}_{k-1}$ by updating to $\underline{S}_k$ using equation (3.15) but with a different choice of c_k . Note also from (3.15) that $\underline{S}_k$ is positive definite for any positive c_k .

Three choices of c_k were experimented with : a constant value $c (= 10^{-4})$, $c_k = -\underline{z}_k^T \underline{g}_{k-1}$, and $c_k = \underline{z}_k^T \underline{z}_k$. The third choice was found to be the most successful. The reason for this is considered to be the fact that this keeps the numerator and denominator of (3.15) of the same order of magnitude; this is especially helpful on approaching convergence when the correction terms become small.

Carrying out a similar analysis for the Davidon-Fletcher-Powell matrix updating scheme, we find that the condition equivalent to $\underline{y}_k^T \underline{g}_{k-1} < 0$ for positive definiteness is that $\underline{p}_k^T \underline{g}_k > 0$. This is satisfied for all positive α_{k-1} if the function is convex along $\underline{p}_k$ (excluding the possibility that $\underline{g}_k = 0$), but in any case it is certainly satisfied at the minimum (since $\underline{p}_k^T \underline{g}_k = 0$ there, and $\underline{S}_{k-1}$ is positive definite). Thus, this method does not suffer from the possibility of failure.

3.5.2 Search Algorithms

It is not difficult to show that conjugate directions are generated if one does go to the minimum along each search direction, provided that the matrix updating scheme does not fail⁽³⁰⁾. Also, Method B and the Fletcher-Powell method will generate identical paths on quadratic functions if they both start with the same initial $\underline{S}_0$. Thus one search algorithm tried was that of searching for the minimum along each direction. This also enabled us to compare the matrix updating scheme itself with that of Davidon-Fletcher-Powell.

One way of avoiding fruitless computation is to step $\alpha_{k-1} = 1$, and test the sign of $\underline{p}_k^T \underline{g}_k$ there. If $\underline{p}_k^T \underline{g}_k > 0$ then the function will have a minimum in the range $0 < \alpha_{k-1} < 1$ and the condition $\underline{y}_k^T \underline{g}_{k-1} < 0$ will be satisfied there. If $\underline{p}_k^T \underline{g}_k \leq 0$ at $\alpha_{k-1} = 1$, the function will normally have decreased and the minimum occurs at $\alpha_{k-1} \geq 1$. In this case it would seem sensible to test the condition $\underline{y}_k^T \underline{g}_k < 0$ there and, if they are not satisfied, to reset $\underline{S}_k$ immediately, since further analysis⁽³⁰⁾ shows that changing α_{k-1} is unlikely to have any effect.

A third algorithm which suggests itself is to discard all reference to conjugate directions and to merely satisfy the conditions required for

overall convergence. Indeed, the method is quadratically convergent for steps of arbitrary length and direction, so that obtaining conjugacy is superfluous anyway. Condition (3.25), bounding the function decrease, is in general a much less stringent requirement than that of going to the minimum⁽³⁰⁾, so the indicated course of action is to satisfy this, and then test (3.24) and $\underline{y}_k^T \underline{g}_{k-1} < 0$ for stability.

The three algorithms are described more explicitly below. Preliminary numerical experiments showed that the cubic minimization method of Fletcher and Powell was sufficiently powerful to justify the computational expense of calculating gradients. Also, a choice of initial step-length offered by them was found to be useful :

$$\alpha_{k-1} = \min \left\{ 1.0, -2 (f(\underline{x}) - f \text{ L.B.}) / \underline{g}_k^T \underline{p}_k \right\} \quad (3.38)$$

where $f \text{ L.B.}$ is the estimated lower bound on the function, which is often zero.

Davidon's recent variance algorithm⁽²⁷⁾ uses the same matrix updating scheme as (3.17), and alters the value of c_k to keep $\underline{S}_k$ positive definite. A fixed steplength of $\alpha_{k-1} = 1$ is used, and if this results in a function increase, the matrix is updated but the step is rejected, so that $\underline{x}_k = \underline{x}_{k-1}$. However, it is felt that this course of action is not desirable, since a function increase at $\alpha_{k-1} = 1$ indicates that an interpolation can be made yielding both a function decrease and a stable matrix $\underline{S}_k$.

Algorithm B.1. (First Conjugate-Direction Method)

(i) Make an initial step according to equation (3.22) with α_{k-1} given by equation (3.38).

(ii) Use cubic minimization to find the minimum.

- (iii) If $\|\underline{g}_k\| \leq \epsilon$ proceed to the local search for confirmation of a minimum.
- (iv) Test $\alpha_{k-1} \geq \delta$ and $\|\underline{S}_{k-1} \underline{g}_{k-1}\| / \|\underline{g}_{k-1}\| \geq \delta$; rescale α_{k-1} and $\underline{S}_{k-1}$ if necessary.
- (v) Test $|c_k| \geq \delta \underline{z}_k^T \underline{z}_k$ and $\underline{z}_k^T \underline{g}_{k-1} / c_k \leq -\delta$; if either fails reset $\underline{S}_k$, but otherwise update $\underline{S}_{k-1}$ to $\underline{S}_k$ using equation (3.17).

Algorithm B.2.(Second Conjugate-Direction Method)

- (i) Make an initial step according to equation (3.22) with $\alpha_{k-1} = 1$.
- (ii) If $\|\underline{g}_k\| \leq \epsilon$ proceed to the local search for confirmation of a minimum.
- (iii) Test $\underline{z}_k^T \underline{g}_{k-1} > \delta$ and if successful go to step (iv). Otherwise test $|c_k| \geq \delta \underline{z}_k^T \underline{z}_k$ and $\underline{z}_k^T \underline{g}_{k-1} / c_k \leq -\delta$; if either fails reset $\underline{S}_k$ and return to (i) for the next step, but otherwise go to step (vi).
- (iv) Use cubic minimization to find the minimum.
- (v) If $\|\underline{g}_k\| \leq \epsilon$ proceed to the local search for confirmation of a minimum.
- (vi) Test $\alpha_{k-1} \geq \delta$ and $\|\underline{S}_{k-1} \underline{g}_{k-1}\| / \|\underline{g}_{k-1}\| \geq \delta$; rescale α_{k-1} and $\underline{S}_{k-1}$ if necessary.
- (vii) Update $\underline{S}_{k-1}$ to $\underline{S}_k$ using equation (3.17).

Algorithm B.2a

- As Algorithm B.2 but with the initial α_{k-1} given by equation (3.38). In this case cubic minimization may generate an $\alpha_{k-1} \geq 1$ and if this occurs we set $\alpha_{k-1} = 1$ and return to (iii).

Algorithm B.3 (Ensuring Convergence)

- (i) Set $\alpha_{k-1} = 1$.
- (ii) Make a step according to equation (3.22).
- (iii) If $\|\underline{g}_k\| \leq \epsilon$ proceed to the local search for confirmation of the minimum.
- (iv) Test condition (3.25). If not satisfied make one step of the cubic minimization algorithm, or if this has converged halve α_{k-1} , and return to (ii).
- (v) Test $\alpha_{k-1} \geq \delta$ and $\|\underline{S}_{k-1} \underline{g}_{k-1}\| / \|\underline{g}_{k-1}\| \geq \delta$; rescale α_{k-1} and $\underline{S}_{k-1}$ if necessary.
- (vi) Test $|c_k| \geq \delta \underline{z}_k^T \underline{z}_k$ and $\underline{z}_k^T \underline{g}_{k-1} / c_k \leq -\delta$; if either fails reset $\underline{S}_k$ but otherwise update $\underline{S}_{k-1}$ to $\underline{S}_k$ using equation (3.17).

Algorithm B.3a

- As Algorithm B.3 but with the initial α_{k-1} in (i) given by equation (3.38).

3.6 Numerical Examples

A number of problems appearing in the literature are specifically designed to be difficult to solve, and have evolved as a standard set for comparison of minimization methods.

Example 1⁽⁶⁾ minimize $f(\underline{x}) = 100 (x_1^2 - x_2)^2 + (1 - x_1)^2$

starting pt. (-1.2, 1.0).

This is a narrow curving banana-shaped valley, with the starting point at one end and the minimum at the other.

Example 2⁽⁶⁾ minimize $f(\underline{x}) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2$
 $+ (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$

starting pt. (3, -1, 0, 1).

This quartic has a singular Hessian at the minimum. It thus presents a difficult test for methods which build up an approximation to the inverse of the Hessian.

Example 3⁽⁶⁾ minimize $f(\underline{x}) = 100 \left[(x_3 - 10\theta)^2 + (r - 1)^2 \right] + x_3^2$

where $2\pi\theta = \tan^{-1} (x_2/x_1) - \pi/2 \leq 2\pi\theta \leq 3\pi/2$

and $r = (x_1^2 + x_2^2)^{1/2}$

starting pt. (-1, 0, 0).

This function has a steep-sided helical valley, winding in the x_3 direction with pitch 10 and radius 1.

Example 4 ⁽³³⁾ minimize $f(\underline{x}) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2 + 90(x_3^2 - x_4)^2$
 $+ (1 - x_3)^2 + 10.1 \left[(x_2 - 1)^2 + (x_4 - 1)^2 \right]$
 $+ 19.8 (x_2 - 1)(x_4 - 1)$

starting pt. $(-3, -1, -3, -1)$

This is a four-dimensional banana-shaped valley, with characteristic steep sides and high curvature. Around the 'middle' there is extreme curvature with little function decrease along the valley floor.

Example 5 ⁽⁶⁾ minimize $f(\underline{x}) = \sum_{i=1}^n \left[E_i - \sum_{j=1}^n (A_{ij} \sin x_j + B_{ij} \cos x_j) \right]^2$

i.e. solve the set of simultaneous non-linear equations

$$\sum_{j=1}^n (A_{ij} \sin x_j + B_{ij} \cos x_j) = E_i \quad i = 1 \text{ --- } n$$

The matrix elements of $\underline{A}$ and $\underline{B}$ were generated as random integers between ± 100 , and the values of the variables, x_i , $i = 1 \text{ --- } n$, were generated randomly between $\pm \pi$. For these values the right-hand sides, E_i , were calculated, and the starting point taken as $(x_i + 0.1 \delta_i)$ where the δ_i 's were also generated as random numbers between $\pm \pi$. The experiments were repeated a number of times for $n = 5, 10$, and 20 .

The examples were solved on an IBM 7094, using double precision (16 decimal) arithmetic. (Some comparative results in single precision showed little change). The constants appearing in section (3.4) and (3.5) for both Method A and Method B, δ and ϵ , were taken as 10^{-8} and 10^{-4} respectively. The cubic minimization method of Fletcher and Powell was used in all the tests, and when searching for a minimum along a line the criterion for convergence was that $\left| \underline{g}_k^T \underline{p}_k \right| \leq \epsilon$ or $\Delta \alpha \left\| \underline{s}_{k-1} \underline{g}_{k-1} \right\| \leq \epsilon$ where $\Delta \alpha$ is the change in α_{k-1} for one iteration of the search algorithm.

Method A

This method was tried on the first four examples. The results of these were sufficient to indicate that it was considerably inferior to Method B, so that further investigation was not warranted. Two alternative search strategies were tested, these involving a small change in step 2 of the procedure described in section 3.4. :

Algorithm A.1 : The minimum along each search direction was sought.

Algorithm A.2 : The search along each direction was terminated when condition (3.25) was satisfied.

For purposes of comparison two similar algorithms using the Davidon-Fletcher-Powell matrix updating scheme were tried :

1. A programme was written in which the minimum along each direction was sought, the approximating matrix being updated using equation (1.13).
2. The other programme used was subroutine VAOIA, supplied by M.J.D. Powell, which is similar to Algorithm A.2 in that it merely seeks a function decrease.

The results of these tests are shown in Table 1 of Appendix 1.

Method B

The three algorithms given in section (3.5) were each tried on the five examples. Two different reset schemes, when the condition

$\underline{y}_k^T \underline{E}_k < 0$ failed, were tried :

Reset 1 : Set $\underline{S}_k = \underline{I}$

Reset 2 : Update $\underline{S}_{k-1}$ to $\underline{S}_k$ using (3.15) with $c_k = \underline{z}_k^T \underline{z}_k$.

The results for the first four examples are shown in Table 2 of Appendix 1. Also in this appendix are Table 3, which shows the results on Example 5, and Tables 4 - 7, which compare the progress of Algorithms B.3 and B.3a (using Reset 2) on the first four examples.

Discussion of Results

It is immediately evident from these results that Method A behaves very poorly in comparison to Method B. The main difference between these methods is that the correction term in Method B is fixed by symmetry, whereas the choice of correction term in Method A is very flexible. It would seem that this arbitrary nature of $\underline{S}_k$ contributes in large measure to the downfall of the method when it is applied to general functions. It is interesting to note the same pattern of behaviour in both Method A and the Fletcher-Powell method : the second search strategy, of merely seeking a function decrease along each direction, produced an improvement in all examples except the fourth, where it failed badly.

The performance of Method B was most encouraging, especially Algorithms B.3 and B.3a. Comparison of the performance of B.3 and B.3a in Tables 2 and 4 - 7 of Appendix 1 shows an interesting phenomenon. B.3a, giving a choice of initial step-length, shows an improvement over B.3 as expected in all examples except 1 and 4 where the functions are characterized by narrow curving valleys. It is evident that the large initial step in B.3 oversteps the minimum, and hence the path zig-zags easily around the bends, whilst the smaller step-size in B.3a results in a path which creeps around on one side of the valley.

Of the three comparable conjugate-direction methods Algorithm B.2a was generally the best, with B.1 usually giving a slight improvement over the Fletcher-Powell (Algorithm 1) method; however, this general trend is reversed in Example 5.

An interesting phenomenon observed in Example 5 was that progress often went in cycles of around n steps; n steps with small function decrease would be made, followed by one or more steps with large function decrease. Often a reset was then required. Presumably, the n steps would build up curvature information, producing a large improvement and leading to a region where this information becomes invalid.

Reset 2 proved to be generally better than Reset 1, as would be expected since only the current step is rejected with Reset 2 rather than the whole matrix as with Reset 1. In fact the need to reset did not occur as often as anticipated. No rescaling of the $\underline{S}_k$ was necessary at any time with any of the methods.

The conclusion to be drawn from these results is that Algorithm B.3a with Reset 2 gives the best all-round performance. However, if it is known that the function is likely to exhibit a curving valley, as often occurs for example in penalty function formulations of problems with non-linear constraints, it is probably worth suppressing the choice of initial step-length and using Algorithm B.3.

3.7 Numerical Evaluation of Derivatives

Stewart⁽³¹⁾ has recently published an algorithm for use with the Fletcher-Powell method when gradients of the function cannot be calculated analytically. He uses the second-order information in the approximating matrix, $\underline{S}_k$, to estimate the relative effects of rounding error and truncation error, using these estimates to determine the step size for calculating gradients by differencing.

The optimal step size is that for which truncation error and rounding error are of the same order of magnitude. On the basis of the estimated magnitudes of these errors the algorithm also chooses between using first-order differences or second-order differences in calculating the gradient numerically.

This algorithm was tried on Method B, the only modification being to allow for the change in matrix-updating formulae. The algorithm, as modified, is summarized below; for the arguments behind the calculations, reference is drawn to Stewart's paper⁽³¹⁾:

Input data : ϵ_f = estimated rounding error in the function calculation. ϵ_m = step size for calculating gradients initially (or on Reset 1).

Steps :

1. Update $\underline{u}$ = vector of diagonal elements of $\underline{S}_{k-1}^{-1}$, using :

$$\underline{S}_k^{-1} = \underline{S}_{k-1}^{-1} - \frac{\frac{w_k w_k^T}{\underline{q}_k^T \underline{z}_k + \underline{w}_k^T \underline{z}_k}}{\underline{q}_k^T \underline{z}_k + \underline{w}_k^T \underline{z}_k}$$

where $\underline{z}_k = \underline{p}_k - \underline{S}_{k-1} \underline{q}_k$

and $\underline{w}_k = (1 - \alpha_{k-1}) \underline{g}_{k-1} - \underline{g}_k$

2. For each component (g_i, x_i) of vectors $\underline{g}_{k-1}$, $\underline{x}_k$ and function value $f = f(\underline{x}_k)$, calculate :

$$(i) \quad \epsilon = \min \left\{ \epsilon_f, \left| g_i \right| \cdot \left| x_i \right| \cdot \epsilon_m / \left| f \right| \right\}$$

(ii) If $g_i^2 \geq \left| u_i \right| \cdot \left| f \right| \cdot \epsilon$ calculate

$$\delta' = 2 \left(\left| f \right| / \left| u_i \right| \right)^{1/2}$$

$$\delta = \delta' \left[1 - \left| u_i \right| \delta' / (3 \left| u_i \right| \cdot \delta' + 4 \left| g_i \right|) \right]$$

Otherwise, calculate

$$\delta' = 2 \left(\left| f \right| \left| g_i \right| / u_i^2 \right)^{1/3}$$

$$\delta = \delta' \left[1 - 2 \left| g_i \right| / (3 u_i \delta' + 4 \left| g_i \right|) \right]$$

(iii) If $\frac{1}{2} \left| u_i \right| \cdot \delta / \left| g_i \right| < 10^{-2}$ use first differences to calculate the i^{th} component of $\underline{g}_k$ with step size δ . Otherwise, use second differences, choosing δ as the positive root of :

$$\frac{1}{2} \left| u_i \right| \delta^2 - \left| g_i \right| \delta - 10^2 \cdot \left| f \right| \cdot \epsilon = 0$$

The cubic minimization method of Fletcher and Powell was used for the linear searches, the directional derivatives being calculated numerically (along the line).

Preliminary results were not encouraging, and an alternative scheme tried was to use the algorithm to produce optimal step-sizes, but use second-differences throughout.

The results of both of the schemes on Examples 1 - 4 are shown in Table 8 of Appendix 1. (Algorithm B.3 was used on Examples 1 and 4, and Algorithm B.3a used on Examples 2 and 3). The results are not impressive, and while it may be concluded that the algorithm does produce convergence, there is evidently scope for large improvement upon these results.

CHAPTER 4

MINIMIZATION WITH CONSTRAINTS4.1 The Projection Operator

We now consider the problem :

$$\text{minimize } f^0(\underline{x}) \quad (4.1)$$

$$\text{subject to } f^j(\underline{x}) \leq 0 \quad 1 \leq j \leq m^* \quad (4.2)$$

If the constraints are linear, they can be written in the form :

$$(\underline{g}^j)^T \underline{x} - b_j \leq 0 \quad 1 \leq j \leq m^* \quad (4.2a)$$

(We assume that the equations have been scaled so that the vectors $\underline{g}^j$, $1 \leq j \leq m^*$, are normalised.)

Suppose that $f^0(\underline{x})$ is a strictly convex quadratic function and that the constrained minimum point lies on the intersection of m linearly independent hyperplanes given by a matrix equation of the form :

$$\underline{G}_m^T \underline{x} - \underline{b} = 0 \quad (4.3)$$

This intersection forms an $(n-m)$ dimensional subspace orthogonal to the m -dimensional subspace spanned by the normals, $\underline{g}^j$, $1 \leq j \leq m$, which form the columns of $\underline{G}_m$.

The minimum point in the constraint space is characterized by a gradient orthogonal to this space :

$$\underline{g}^0 = \underline{G}_m \underline{\lambda} \quad (4.4)$$

where $\underline{\lambda}$ is an arbitrary m -vector.

Suppose we stepped to the unconstrained minimum point $\hat{\underline{x}}$ from any point $\underline{x}_{k-1}$

$$\hat{\underline{x}} - \underline{x}_{k-1} = \hat{\underline{p}} = -\underline{H}^{-1} \underline{g}_{k-1}^0 \quad (4.5)$$

Now this point, $\hat{\underline{x}}$, transgresses the set of m constraints associated with (4.3), and we therefore seek $\bar{\underline{x}}$ to satisfy (4.4), and also :

$$\underline{G}_m^T (\bar{\underline{x}} - \underline{x}_{k-1}) + \underline{f}_{k-1} = 0 \quad (4.6)$$

where $\underline{f}_{k-1} = \underline{f}^j(\underline{x}_{k-1}) \quad 1 \leq j \leq m \quad (4.7)$

Now, $(\bar{\underline{x}} - \underline{x}_{k-1}) = \underline{H}^{-1}(\underline{G}_m^0 - \underline{E}_{k-1}^0) \quad (4.8)$

and using (4.4)

$$\bar{\underline{x}} - \underline{x}_{k-1} = \underline{H}^{-1}(\underline{G}_m \underline{\lambda} - \underline{E}_{k-1}^0) \quad (4.9)$$

$\underline{\lambda}$ is determined from (4.6) :

$$\underline{G}_m^T (\bar{\underline{x}} - \underline{x}_{k-1}) = (\underline{G}_m^T \underline{H}^{-1} \underline{G}_m) \underline{\lambda} - \underline{G}_m^T \underline{H}^{-1} \underline{E}_{k-1}^0 = -\underline{f}_{k-1} \quad (4.10)$$

$$\underline{\lambda} = (\underline{G}_m^T \underline{H}^{-1} \underline{G}_m)^{-1} (\underline{G}_m^T \underline{H}^{-1} \underline{E}_{k-1}^0 - \underline{f}_{k-1}) \quad (4.11)$$

Note that the inverse exists, since $\underline{H}$ is positive definite.

Using (4.9), the step, $\bar{\underline{p}}$, is thus given by :

$$\bar{\underline{p}} = \bar{\underline{x}} - \underline{x}_{k-1} = \underline{H}^{-1}(\underline{G}_m (\underline{G}_m^T \underline{H}^{-1} \underline{G}_m)^{-1} (\underline{G}_m^T \underline{H}^{-1} \underline{E}_{k-1}^0 - \underline{f}_{k-1}) - \underline{E}_{k-1}^0) \quad (4.12)$$

and using (4.5) :

$$\bar{\underline{p}} = \hat{\underline{p}} - \underline{H}^{-1} \underline{G}_m (\underline{G}_m^T \underline{H}^{-1} \underline{G}_m)^{-1} (\underline{G}_m^T \hat{\underline{p}} + \underline{f}_{k-1}) \quad (4.13)$$

Note that, if $\underline{x}_{k-1}$ is in the subspace, then $\underline{f}_{k-1} = \underline{0}$; in which case equation (4.12) reduces to :

$$\bar{\underline{p}} = \underline{P} \hat{\underline{p}} = -\underline{P} \underline{H}^{-1} \underline{E}_{k-1}^0 \quad (4.14)$$

where $\underline{P} = \{ \underline{I} - \underline{H}^{-1} \underline{G}_m (\underline{G}_m^T \underline{H}^{-1} \underline{G}_m)^{-1} \underline{G}_m^T \}$ (4.15)

$\underline{P}$ is idempotent and is therefore a projection operator, which projects any vector into the $(n-m)$ dimensional subspace formed by the intersection of the m constraining hyperplanes.

If $\underline{H}^{-1}$ is replaced by its current approximation, $\underline{S}_{k-1}$, and $\underline{S}_{k-1}$ is such that the inverse $(\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1}$ exists, then equations (4.11) and (4.13) become :

$$\underline{p}_k = \hat{\underline{p}}_k + \underline{S}_{k-1} \underline{G}_m \lambda_{m,k-1} \quad (4.16)$$

$$\text{where } \lambda_{m,k-1} = (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_{k-1} \underline{E}_{k-1}^0 - \underline{f}_{k-1}) \quad (4.17)$$

$$\text{and } \hat{\underline{p}}_k = - \underline{S}_{k-1} \underline{E}_{k-1}^0 \quad (4.18)$$

Equations (4.14) and (4.15) become :

$$\underline{p}_k = - \alpha_{k-1} \underline{P}_{m,k-1} \underline{S}_{k-1} \underline{E}_{k-1}^0 \quad (\text{with } \alpha_{k-1} = 1) \quad (4.19)$$

$$\text{where } \underline{P}_{m,k-1} = \left\{ \underline{I} - \underline{S}_{k-1} \underline{G}_m (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \underline{G}_m^T \right\} \quad (4.20)$$

Note that $\underline{P}_{m,k-1}$ also projects any vector into the intersection of constraining hyperplanes. $\underline{P}_{m,k-1}$ becomes an orthogonal projection operator analogous to Rosen's if we set $\underline{S}_{k-1} = \underline{I}$.

Note that this projection operator is not symmetric. While its range is the subspace formed by the intersection of constraining hyperplanes, its null space is the space spanned by the columns of $(\underline{S}_k \underline{G}_m)$, not $\underline{G}_m$. For symmetric $\underline{S}_k$, one could describe the operator as a variable-metric projection operator as it is orthogonal in the metric $\underline{S}_k^{-1}$, which if $\underline{S}_k$ were a valid approximation to $\underline{H}^{-1}$ would be the natural metric of the quadratic form.

Note also that $\underline{S}_k$ need only have the required equivalence to $\underline{H}^{-1}$ within a subspace, namely that $\underline{S}_k \underline{H} \underline{u} = \underline{u}$ for every vector $\underline{u}$ in the subspace, without $\underline{S}_k$ being equal to $\underline{H}^{-1}$. That this must be so will be obvious in Chapter 5, where it is shown that only $n-m$ steps, not n , are needed to predict the minimum in an $n-m$ dimensional subspace. But this fact also becomes apparent in Method A, since for this method $\underline{S}_k$ is not usually symmetric at a constrained minimum, but does indeed become symmetric at an unconstrained minimum.

4.2 Updating Required for $\underline{S}_{k-1}$

At each step, $\underline{S}_{k-1}$ is updated to $\underline{S}_k$ by Methods A or B of Chapter 3. When constraints are present, we must then also update $(\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1}$. The two methods show advantages and disadvantages at this stage, each requiring different treatment.

Method A

With this method we can use the degrees of freedom in the choice of $\underline{y}_k$ to make unnecessary any recursive updating of these matrices.

Each time we add a new constraint to the basis, we not only effectively put $\underline{q}_k$ in column j of $\underline{Q}_{k-1}$ by updating $\underline{R}_{k-1}$, but we also put $\underline{g}^i$ (corresponding to the new constraint i) in some other column of $\underline{Q}_k$ after the next step using (3.32). This involves an extra recursion on $\underline{R}_k$, but note that this is required only when we add a new constraint to the basis, and not at every step. On dropping a constraint, (say i'), from the basis, the column of $\underline{Q}_{k-1}$ containing $\underline{g}^{i'}$ simply becomes eligible for replacement by $\underline{q}_k$, along with the others.

This procedure ensures that :

$$\underline{y}_k^T \underline{G}_m = 0 \quad (4.21)$$

Hence, from (3.14) :

$$\underline{S}_k \underline{G}_m = \underline{S}_{k-1} \underline{G}_m \quad (4.22)$$

$$\text{so that } (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} = (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \quad (4.23)$$

$$\text{and also } \underline{P}_{m,k} = \underline{P}_{m,k-1}$$

Note that although the constraint normals occupy m columns of $\underline{Q}_k$, this does not prevent quadratic convergence in a subspace, since at most $(n-m)$ steps are needed to predict the minimum point. Further, from (3.14) and (4.22), we have $\underline{S}_k \underline{q}_j = \underline{S}_{k-1} \underline{q}_j$ for any previous vector, $\underline{q}_j$ implicit in $\underline{R}_{k-1}$ and orthogonal to the subspace. Also, we will not generate a vector $\underline{q}_k$ which is linearly dependent on the columns of $\underline{G}_m$, for this would imply $\underline{p}_k^T \underline{H} \underline{p}_k = 0$. This is impossible since $\underline{H}$ is positive-definite.

Method B

Using this method of updating $\underline{S}_{k-1}$, $\underline{y}_k$ is uniquely determined and we cannot impose the conditions (4.21). We must therefore use recursion formulae. However, since the correction applied to $\underline{S}_{k-1}$ is in the form of a simple product, we can update the inverse moment matrix without difficulty.

$$\begin{aligned}
 (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} &= \left\{ \underline{G}_m^T \underline{S}_{k-1} \underline{G}_m + \underline{G}_m^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) \underline{y}_k^T \underline{G}_m \right\}^{-1} \\
 &= (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} - \frac{\underline{v}_k \underline{y}_k^T \underline{G}_m (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1}}{1 + \underline{y}_k^T \underline{G}_m \underline{v}_k} \quad (4.24)
 \end{aligned}$$

$$\text{where} \quad \underline{v}_k = (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \underline{G}_m^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) \quad (4.25)$$

Note that this recursion must be applied at every step.

4.3 Adding and Dropping Constraints

For updating the inverse moment matrix, $(\underline{G}_{-m}^T \underline{S}_{-k} \underline{G}_{-m})^{-1}$, we use recursion formulae analogous to those used by Rosen :

(i) Adding Constraint to Basis

$$(\underline{G}_{-m+1}^T \underline{S}_{-k} \underline{G}_{-m+1}) = \underline{A} = \begin{bmatrix} \underline{A}_{11} & \underline{A}_{12} \\ \underline{A}_{21} & \underline{A}_{22} \end{bmatrix} \quad (4.26)$$

where $\underline{A}_{11} = \underline{G}_{-m}^T \underline{S}_{-k} \underline{G}_{-m}$

$$\underline{A}_{12} = \underline{G}_{-m}^T \underline{S}_{-k} \underline{G}^{m+1} \quad (4.27)$$

$$\underline{A}_{21} = (\underline{G}^{m+1})^T \underline{S}_{-k} \underline{G}_{-m}$$

$$\underline{A}_{22} = (\underline{G}^{m+1})^T \underline{S}_{-k} \underline{G}^{m+1}$$

Write : $\underline{A}_0 = \underline{A}_{22} - \underline{A}_{21} \underline{A}_{11}^{-1} \underline{A}_{12}$, which is a scalar. (4.28)

Hence :

$$(\underline{G}_{-m+1}^T \underline{S}_{-k} \underline{G}_{-m+1})^{-1} = \underline{B} = \begin{bmatrix} \underline{B}_{11} & \underline{B}_{12} \\ \underline{B}_{21} & \underline{B}_{22} \end{bmatrix} \quad (4.29)$$

where $\underline{B}_{11} = \underline{A}_{11}^{-1} + \underline{A}_{11}^{-1} \underline{A}_{12} \underline{A}_0^{-1} \underline{A}_{21} \underline{A}_{11}^{-1}$

$$\underline{B}_{12} = - \underline{A}_{11}^{-1} \underline{A}_{12} \underline{A}_0^{-1} \quad (4.30)$$

$$\underline{B}_{21} = - \underline{A}_0^{-1} \underline{A}_{21} \underline{A}_{11}^{-1}$$

$$\underline{B}_{22} = \underline{A}_0^{-1}$$

The only inverse matrix required is :

$$\underline{A}_{11}^{-1} = (\underline{G}_{-m}^T \underline{S}_{-k} \underline{G}_{-m})^{-1} , \text{ which is known.}$$

Assuming that $\underline{g}^{m+1}$ is linearly independent of the columns of $\underline{G}_m$, we must still show that the new inverse always exists,

$$\text{i.e.} \quad \det \begin{vmatrix} \underline{G}_{m+1}^T & \underline{S}_k & \underline{G}_{m+1} \end{vmatrix} \neq 0 \quad (4.31)$$

Note that the result follows immediately for Method B of updating $\underline{S}_{k-1}$, as $\underline{S}_k$ remains positive definite. (Note also that $\underline{A}_{12} = \underline{A}_{21}^T$ for Method B.)

This is not so for Method A, but since we have a certain degree of arbitrariness in the choice of $\underline{S}_k$ we can take explicit precautions to satisfy (4.31). By a Cauchy expansion of the determinant :

$$\det \begin{vmatrix} \underline{G}_{m+1}^T & \underline{S}_k & \underline{G}_{m+1} \end{vmatrix} = A_0 \cdot \det \begin{vmatrix} \underline{G}_m^T & \underline{S}_k & \underline{G}_m \end{vmatrix}$$

$$\text{where} \quad A_0 = (\underline{g}^{m+1})^T \underline{P}_{m,k} \underline{S}_k \underline{g}^{m+1} \quad (4.32)$$

Now if it should happen that $A_0 = 0$, we then make a small step, without adding the hyperplane, and update $\underline{S}_k$ again. We then have :

$$A_0 = (\underline{g}^{m+1})^T \underline{P}_{m,k+1} \underline{S}_k \underline{g}^{m+1} + (\underline{g}^{m+1})^T \underline{P}_{m,k+1} (\underline{p}_{k+1} - \underline{S}_k \underline{q}_{k+1}) \underline{y}_{k+1}^T \underline{g}^{m+1}$$

$$= c_1 + c_2 \underline{y}_{k+1}^T \underline{g}^{m+1} \quad (4.33)$$

where $c_1 = 0$, and c_2 is invariant on further choice of $\underline{y}_{k+1}$, since $\underline{P}_{m,k+1} = \underline{P}_{m,k}$ for Method A.

If the determinant is again zero for this new step, we can impose a further condition on $\underline{y}_{k+1}$:

$$\underline{y}_{k+1}^T \underline{g}^{m+1} = 1 \quad (4.34)$$

However, in the further event that $c_2 = 0$ over a range of α_k , we have a degenerate case. This is only a remote possibility and the simplest procedure seems to be to reset $\underline{S}_k$ to a new positive definite matrix, for example the identity matrix.

(ii) Dropping Constraint from Basis

$$(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} = \underline{A}^{-1} = \underline{B} = \begin{bmatrix} \underline{B}_{11} & \underline{B}_{12} \\ \underline{B}_{21} & \underline{B}_{22} \end{bmatrix} \quad (4.35)$$

$$\begin{aligned} \text{Solving for } \underline{A}_{11}^{-1} &= (\underline{G}_{m-1}^T \underline{S}_k \underline{G}_{m-1})^{-1} \\ &= \underline{B}_{11} - \underline{B}_{12} \underline{B}_{22}^{-1} \underline{B}_{21} \end{aligned} \quad (4.36)$$

It can be shown, by using a permutation matrix, that this formula also applies for the j^{th} constraint to be dropped, with the m^{th} and j^{th} rows and columns interchanged.

4.4 Kuhn-Tucker Theory and the Projection Operator

As is to be expected, solution of the Lagrangian saddle-point problem and constrained minimization with the projection operator have many similarities, both algebraically and geometrically. In fact, if we consider only linear constraints, then it is not difficult to show that they are equivalent.

Consider the Lagrangian function :

$$L(\underline{x}, \underline{\lambda}) = f^0(\underline{x}) - \sum_{j=1}^m \lambda_j f^j(\underline{x}) \quad (4.37)$$

$$L(\underline{x}_k, \underline{\lambda}) = L(\underline{x}_{k-1}, \underline{\lambda}) + \underline{g}^L(\underline{x}_{k-1})^T \underline{p}_k + \frac{1}{2} \underline{p}_k^T \underline{H}^L(\underline{x}_{k-1} + \theta \underline{p}_k) \underline{p}_k$$

$$0 \leq \theta \leq 1 \quad (4.38)$$

$$\text{where } \underline{g}^L(\underline{x}_{k-1}) = \underline{g}^0 - \sum_{j=1}^m \lambda_j \underline{g}^j$$

$$\text{and } \underline{H}^L(\underline{x}_{k-1}) = \underline{H}^0(\underline{x}_{k-1}) \text{ when the constraints are linear.}$$

The solution point, $(\bar{\underline{x}}, \bar{\underline{\lambda}})$, of the saddle point problem :

$$\min_{\underline{x}} \max_{\underline{\lambda} \leq 0} L(\underline{x}, \underline{\lambda})$$

must satisfy the necessary conditions :

$$(i) \quad \frac{\partial L}{\partial \underline{x}}(\bar{\underline{x}}, \bar{\underline{\lambda}}) = 0$$

$$(ii) \quad \frac{\partial L}{\partial \underline{\lambda}}(\bar{\underline{x}}, \bar{\underline{\lambda}}) \geq 0$$

$$(iii) \quad \bar{\underline{\lambda}}^T \frac{\partial L}{\partial \underline{\lambda}}(\bar{\underline{x}}, \bar{\underline{\lambda}}) = 0$$

(i) To obtain $\frac{\partial L}{\partial \underline{x}}(\underline{\bar{x}}, \underline{\bar{\lambda}}) = 0$, we must step

$$\begin{aligned}\underline{\bar{x}} &= \underline{x}_{k-1} - (\underline{H}^L)^{-1} \underline{E}_k^L \\ &= \underline{x}_{k-1} - (\underline{H}^L)^{-1} (\underline{E}_k^0 - \sum_{j=1}^m \lambda_j \underline{E}^j)\end{aligned}$$

Hence we obtain :

$$\underline{x}_k = \underline{x}_{k-1} - \underline{S}_k (\underline{E}_k^0 - \underline{G}_m \underline{\lambda}) \quad (4.39)$$

(ii) As the Lagrangian is linear in $\underline{\lambda}$, $\frac{\partial L}{\partial \underline{\lambda}} \geq 0$ reduces to a set of inequalities in $\underline{x}$:

$$f^j(\underline{x}) \leq 0 \quad j = 1 \dots m^* \quad (4.40)$$

which is of course the same as (4.2).

It was noted earlier in Chapter 2 that condition (iii) implies that either $\bar{\lambda}_j = 0$ or $f^j(\underline{x}) = 0$, so that constraints not active at $\underline{\bar{x}}$ are ignored. This interpretation is used in combining (4.39) with (4.40) for the current set of active constraints in the basis to give $\underline{\lambda}_{m,k}$ as the solution of :

$$(\underline{G}_m^T \underline{S}_k \underline{G}_m) \underline{\lambda}_{m,k} = \underline{G}_m^T \underline{S}_k \underline{E}_k^0$$

What is interesting, however, is that we may view a projected direction as being an unconstrained step on the Lagrangian function, with the Lagrange multipliers being updated recursively at each step. This may be compared to the methods discussed in section (2.2.2) where the Lagrangian is minimized over the whole space of $\underline{x}$ for a fixed $\underline{\lambda}$. Note also that, using the projection operator, we do not work with the whole space of $\underline{\lambda}$ at each step, those not in the current basis being set to zero.

The Kuhn-Tucker necessary conditions for a constrained overall minimum may be expressed conveniently in the form :

$$\underline{P}_{m,k} \underline{S}_k \underline{E}_k^0 = 0 \quad (4.41)$$

$$f^j(\underline{x}_k) \leq 0 \quad j = 1 \text{ --- } m^x \quad (4.42)$$

$$\lambda^j(\underline{x}_k) \leq 0 \quad j = 1 \text{ --- } m^x \quad (4.43)$$

$$\text{where} \quad \underline{\lambda}_{m,k} = (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} \underline{G}_m^T \underline{S}_k \underline{E}_k^0 \quad (4.44)$$

$$\text{Note that } \underline{P}_{m,k} \underline{S}_k \underline{E}_k^0 = \underline{S}_k (\underline{E}_k^0 - \underline{G}_m \underline{\lambda}_{m,k}) \quad (4.45)$$

Since $\underline{S}_k$ is non-singular and bounded below by construction, equation (4.41) is equivalent to (4.4). Condition (4.41) states that the negative gradient direction must be orthogonal to the set of hyperplanes which satisfy the linearized version of (4.42) as equalities. Condition (4.43) may be interpreted as requiring that the negative gradient direction be within the cone spanned by the constraint normals, or more simply that the negative gradient direction points orthogonally out of the feasible region with respect to all the active constraints. If $\underline{P}_{m,k} \underline{S}_k \underline{E}_k^0 = 0$, but $\lambda_j > 0$ for some constraint j , the negative gradient direction is pointing orthogonally into the feasible region with respect to this constraint. Hence, this constraint must be dropped, and the projection operator then projects into a subspace one dimension higher.

It is in fact not necessary to search in a subspace until the gradient becomes orthogonal to the constraint surfaces before testing. In a manner similar to Rosen⁽⁷⁾ it is possible to detect that a constraint needs to be dropped as the gradient comes within a certain angle to the perpendicular. In other words, one can derive a lower bound for

$\underline{P}_{m,k} \underline{S}_k \underline{E}_k^0$ in terms of each λ_j , below which the constraint j may be dropped :

Dividing the vector $\underline{S}_k \underline{E}_k^0$ into the range and null-space complements of $\underline{P}_{m,k}$, we have :

$$\underline{S}_k \underline{E}_k^0 = \underline{S}_k \underline{G}_m \underline{\lambda} + \underline{P}_{m,k} \underline{S}_k \underline{E}_k^0 \quad (4.46)$$

where $\underline{\lambda}$ is given by (4.44)

$$\text{Hence, } \underline{P}_{m-1,k} \underline{S}_k \underline{E}_k^0 = \underline{P}_{m-1,k} \underline{S}_k \underline{G}_m \underline{\lambda} + \underline{P}_{m-1,k} \underline{P}_{m,k} \underline{S}_k \underline{E}_k^0 \quad (4.47)$$

Now it is easily shown that :

$$\underline{P}_{m-1,k} \underline{S}_k \underline{G}_m \underline{\lambda} = \lambda_m \underline{P}_{m-1,k} \underline{S}_k \underline{E}^m \quad (4.48)$$

$$\text{and } \underline{P}_{m-1,k} \underline{P}_{m,k} \underline{S}_k \underline{E}_k^0 = \underline{P}_{m,k} \underline{S}_k \underline{E}_k^0 \quad (4.49)$$

Hence, (4.47) becomes :

$$\underline{P}_{m-1,k} \underline{S}_k \underline{E}_k^0 = \lambda_m \underline{P}_{m-1,k} \underline{S}_k \underline{E}^m + \underline{P}_{m-1,k} \underline{S}_k \underline{E}_k^0$$

$$\text{so that } \left\| \underline{P}_{m-1,k} \underline{S}_k \underline{E}_k^0 \right\| \geq \lambda_m \left\| \underline{P}_{m-1,k} \underline{S}_k \underline{E}^m \right\| - \left\| \underline{P}_{m-1,k} \underline{S}_k \underline{E}_k^0 \right\| (\lambda_m > 0) \quad \dots (4.50)$$

(Note again that this applies to any constraint, j , to be dropped, with the j^{th} and m^{th} rows and columns interchanged.)

From (4.28) and (4.30) note that the m^{th} diagonal element of $(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1}$, $b_{mm} = A_o^{-1}$

$$\begin{aligned} \text{where } A_o &= (\underline{E}^m)^T \underline{S}_k \underline{E}^m - (\underline{E}^m)^T \underline{S}_k \underline{G}_{m-1} (\underline{G}_{m-1}^T \underline{S}_k \underline{G}_{m-1})^{-1} \underline{G}_{m-1}^T \underline{S}_k \underline{E}^m \\ &= (\underline{E}^m)^T \underline{P}_{m-1,k} \underline{S}_k \underline{E}^m \end{aligned} \quad (4.51)$$

Using Schwarz inequality :

$$\left\| \underline{g}^m \right\| \cdot \left\| \underline{P}_{m-1,k} \underline{S}_k \underline{g}^m \right\| \geq \left| (\underline{g}^m)^T \underline{P}_{m-1,k} \underline{S}_k \underline{g}^m \right| = \frac{1}{|b_{mm}|}$$

Now we have assumed the constraint equations have been normalized so that $\left\| \underline{g}^m \right\| = 1$, hence (4.50) reduces to :

$$\left\| \underline{P}_{m-1,k} \underline{S}_k \underline{g}_k^0 \right\| \geq \lambda_m / |b_{mm}| - \left\| \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 \right\| \quad (4.52)$$

Therefore, if we drop the j^{th} constraint when

$$\left\| \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 \right\| < \frac{1}{2} \lambda_j / |b_{jj}| \quad j = j : \lambda_j / |b_{jj}| \geq \lambda_i / |b_{ii}|, \lambda_i > 0$$

..... (4.53)

then

$$\left\| \underline{P}_{m-1,k} \underline{S}_k \underline{g}_k^0 \right\| > \left\| \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 \right\| \quad (4.54)$$

Thus, the norm of the projected direction is increased on dropping constraint j from the basis when the inequality (4.53) is satisfied.

In the case of Method B it is worth noting that, as in the unconstrained case, positive-definiteness of $\underline{S}_k$ is a sufficient condition to ensure that the projected direction is downhill, i.e. that it makes an acute angle with the negative gradient direction :

$$\text{from (4.45)} \quad (\underline{g}_k^0)^T \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 = (\underline{g}_k^0)^T \underline{S}_k (\underline{g}_k^0 - \underline{G}_m \underline{\lambda}_{m,k})$$

with $\underline{\lambda}_{m,k}$ given by equation (4.44).

$$\begin{aligned} \text{Hence } (\underline{g}_k^0)^T \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 &= (\underline{g}_k^0 - \underline{G}_m \underline{\lambda}_{m,k})^T \underline{S}_k (\underline{g}_k^0 - \underline{G}_m \underline{\lambda}_{m,k}) \\ &\quad + (\underline{G}_m \underline{\lambda}_{m,k})^T \underline{S}_k (\underline{g}_k^0 - \underline{G}_m \underline{\lambda}_{m,k}) \end{aligned}$$

But, using (4.44) we find that the second term on the right-hand side is zero, and for positive definite $\underline{S}_k$ the first term on the right hand side is positive unless $\underline{g}_k^0 = \underline{G}_{-m} \underline{\lambda}_{m,k}$, which is condition (4.4) for a constrained stationary point. We thus have $(\underline{g}_k^0)^T \underline{P}_{-m,k} \underline{S}_k \underline{g}_k^0 > 0$, as required, until a constrained stationary point is reached.

4.5 Constrained Step Strategy for Linear Constraints

4.5.1 Algorithm

This is analogous to Rosen's strategy for linear constraints. If $\underline{x}_k$ is an interior point, we search in the unconstrained direction :

$$\underline{p}_{k+1} = -\alpha_k \underline{s}_k \underline{E}_k^0 \quad (4.55)$$

If $\underline{x}_k$ is on a set of active constraints, we project the unconstrained direction into the intersection of hyperplanes which form the constraint boundaries :

$$\underline{p}_{k+1} = -\alpha_k \underline{P}_{m,k} \underline{s}_k \underline{E}_k^0 \quad (4.56)$$

In this case α_k is chosen as for the unconstrained method unless a new constraint is encountered.

A general step of the algorithm, starting at point $\underline{x}_k$ is as follows :

I Update :

Method A

(i) Compute $\underline{y}_k^T$ and $\underline{R}_k$ from equations (3.12) and (3.13) choosing the column j to be replaced to be the oldest column in $\underline{Q}_{k-1}$ for which

$$\left| \underline{e}_j^T \underline{R}_{k-1} \underline{q}_k \right| \geq \epsilon \quad (\text{excluding the columns containing } \underline{g}^1 \text{ --- } \underline{g}^m).$$

(ii) If a new constraint, i , was incorporated in the basis for the previous step $\underline{p}_k$, put $\underline{g}^i$ in the j^{th} column of $\underline{Q}_k$ by using the recursion formula (3.10) on $\underline{g}^i$ and $\underline{R}_k$, giving $\underline{R}_k^1$.

$$\underline{J}^N = \underline{J}^N \mid \underline{e}_{j^N}^T \underline{R}_k \underline{g}^i \geq \underline{e}_m^T \underline{R}_k \underline{g}^i, \quad m \in J$$

where J is the set of columns occupied by either $\underline{q}_k$, or $\underline{g}^1 \text{ --- } \underline{g}^m$.

(iii) Update $\underline{S}_{k-1}$ using equation (3.14), with $\underline{y}_k^T = \underline{e}_j^T \underline{R}_k$
(or $\underline{y}_k^T = \underline{e}_j^T \underline{R}_k^1$ for step (ii)).

Method B

(i) Check that $\left| \underline{q}_k^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k) \right| > \delta$

(ii) Check for positive-definiteness. Note that this test takes a different form in the presence of linear constraints :

$$\det \left| \underline{S}_k \right| = \det \left| \underline{S}_{k-1} \right| \left\{ \underline{y}_k^T \underline{S}_{k-1}^{-1} \underline{p}_k \right\}$$

for $\underline{y}_k$ given by (3.18).

$$\text{i.e. } \det \left| \underline{S}_k \right| = \det \left| \underline{S}_{k-1} \right| \left\{ \alpha_{k-1} \underline{y}_k^T (\underline{G}_m \underline{\lambda} - \underline{E}_{k-1}^0) \right\}$$

where $\underline{\lambda}$ is given by equation (4.44).

Hence, test that $\underline{y}_k^T (\underline{G}_m \underline{\lambda} - \underline{E}_{k-1}^0) > \delta$, for $\underline{y}_k$ given by (3.18).

(iii) If these tests are satisfied, update $\underline{S}_{k-1}$ using equation (3.14) with $\underline{y}_k$ given by (3.18).

Otherwise, update using $\underline{y}_k$ given by :

$$\underline{y}_k = \frac{(\underline{p}_k - \underline{S}_{k-1} \underline{q}_k)}{(\underline{p}_k - \underline{S}_{k-1} \underline{q}_k)^T (\underline{p}_k - \underline{S}_{k-1} \underline{q}_k)} \quad (4.57)$$

This is preferable to resetting $\underline{S}_k = \underline{I}$, as this would require a resetting and reinversion of the moment matrix.

(iv) Update the inverse moment matrix,

$$(\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \text{ to } (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} , \text{ using equation (4.24).}$$

II Projection :

$$(i) \text{ Form } \underline{\lambda}_{m,k} = (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} \underline{G}_m^T \underline{S}_k \underline{E}_k^o \quad (4.44)$$

$$(ii) \text{ Form } \underline{p}_{k+1} = - \underline{P}_{m,k} \underline{S}_k \underline{E}_k^o = \underline{S}_k (\underline{G}_m \underline{\lambda}_{m,k} \underline{E}_k^o) \quad (4.58)$$

$$(iii) \text{ Form } \beta = \frac{1}{2} \lambda_j / |b_{jj}| \quad j = j \left| \frac{\lambda_j}{|b_{jj}|} \geq \frac{\lambda_i}{|b_{ii}|} \right|, 1 \leq i \leq m$$

..... (4.59)

b_{jj} is the corresponding j^{th} diagonal element of $(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1}$.

Strict equality constraints are excluded from consideration.

(iv) If $\underline{x}_k$ is on the boundary of a new constraint, $m+1$, calculate $\underline{p}_{k+1}^T \underline{E}^{m+1}$ and $(\underline{E}_k^o)^T \underline{E}^{m+1}$. If either is positive, add this constraint to the basis, update $(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1}$ to $(\underline{G}_{m+1}^T \underline{S}_k \underline{G}_{m+1})^{-1}$ using equation (4.29) and put $m = m+1$. Return to step (i).

(v) Stop if $\|\underline{p}_{k+1}\| \leq \xi$ and $\beta \leq \xi$, where ξ is some small number : the Kuhn-Tucker conditions for an overall minimum are satisfied.

(vi) If $\|\underline{p}_{k+1}\| < \beta$, drop the constraint corresponding to λ_j from the basis, update $(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1}$ to $(\underline{G}_{m-1}^T \underline{S}_k \underline{G}_{m-1})^{-1}$ using equation (4.36), and put $m = m-1$. Return to step (i).

(vii) Check that

$$(\underline{E}_k^o)^T \underline{P}_{m,k} \underline{S}_k \underline{E}_k^o \geq \epsilon \cdot \|\underline{E}_k^o\| \cdot \left\| \underline{P}_{m,k} \underline{S}_k \underline{E}_k^o \right\|$$

(For Method A, merely check the magnitude of the left hand side, since a negative α_k is possible). If not satisfied, reset $\underline{S}_k = \underline{I}$, reform and reinvert the moment matrix $(\underline{G}_m^T \underline{G}_m)$. If $\gamma = \|\underline{p}_{k+1}\| / \|(\underline{G}_m \underline{\lambda} - \underline{E}_k^o)\| < \delta$, rescale by dividing the elements of $\underline{S}_k$ and $\underline{p}_{k+1}$ by γ and multiplying the elements of $(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1}$ by γ .

III Search :

(i) For Method A, choose the sign of α_k to make $\alpha_k \underline{p}_{k+1}^T \underline{g}_k^0$ negative.

(ii) Calculate

$$\alpha_k^* = \min_j \left\{ (b_j - (\underline{g}^j)^T \underline{x}_k) / \underline{p}_{k+1}^T \underline{g}^j \right\}$$

$$m+1 \leq j \leq m^*$$

= position of nearest constraint.

(iii) Put $\alpha_k = \min(1.0, \alpha_k^*)$

$$\text{step } \underline{x}^1 = \underline{x}_k + \alpha_k \underline{p}_{k+1}$$

Test, if $f(\underline{x}_k) - f(\underline{x}^1) \geq \delta \cdot \alpha_k \cdot (\underline{g}_k^0)^T \underline{p}_{m,k} \underline{s}_k \underline{g}_k^0$

put $\underline{x}_{k+1} = \underline{x}^1$, otherwise interpolate to adjust α_k until this is satisfied.

(iv) If $|\alpha_k| < \delta$, rescale α_k , $\underline{s}_k$ and $(\underline{g}_m^T \underline{s}_k \underline{g}_m)^{-1}$.

The above description does not include some additional features of the type developed by Rosen. Not all of the results obtained by Rosen generalize to the case of oblique projectors, owing to the lack of symmetry in $\underline{p}_{m,k}$ and the presence of the matrix $\underline{s}_k$. However, it is possible to derive an equivalent set of those results which are essential.

4.5.2 Linear Dependence of Hyperplanes

Rosen has utilised the fact that an orthogonal projection operator provides a convenient test for linear dependence, since it annihilates any vector in the space spanned by the columns of $\underline{G}_m$. If a hyperplane about to be added to the constraint set was linearly dependent on the set of hyperplanes forming the $(n-m)$ -dimensional subspace, then their

normals also would be linearly dependent :

$$\underline{g}^{m+1} = \underline{G}_m \underline{\alpha} \text{ for some non-trivial } \underline{\alpha}.$$

The projection operator $\underline{P}_{m,k}$ is an oblique projector and would not annihilate $\underline{g}^{m+1}$ in this case. However, it is evident that if we rotate $\underline{g}^{m+1}$ by the matrix $\underline{S}_k$ and then project, $\underline{P}_{m,k}$ will annihilate it.

$$\begin{aligned} \underline{P}_{m,k} \underline{S}_k \underline{g}^{m+1} &= (\underline{I} - \underline{S}_k \underline{G}_m (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} \underline{G}_m^T) \underline{S}_k \underline{g}^{m+1} \\ &= \underline{0} \quad \text{if} \quad \underline{g}^{m+1} = \underline{G}_m \underline{\alpha} \end{aligned}$$

Thus, if $\left\| \underline{P}_{m,k} \underline{S}_k \underline{g}^{m+1} \right\| \leq \xi$, $\underline{g}^{m+1}$ is not added to the constraint basis.

4.5.3 Degeneracy

When linearly dependent hyperplanes are present, a degenerate situation may arise as follows.

Suppose the point $\underline{x}_k$ lies on $m+r$ hyperplanes, r of which are linearly dependent on the first m .

(i) If $\left\| \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 \right\| \geq \beta$, $\underline{p}_{k+1}$ is satisfactory,

for note that we have $\underline{G}_m^T \underline{p}_{k+1} = \underline{0}$ and also $(\underline{g}^{m+1} \cdots \underline{g}^{m+r})^T \underline{p}_{k+1} = \underline{0}$ for the linearly dependent subset.

(ii) If $\left\| \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 \right\| \leq \xi$ and $\beta \leq \xi$, the Kuhn-Tucker conditions for an overall minimum are satisfied.

(iii) If $\left\| \underline{P}_{m,k} \underline{S}_k \underline{g}_k^0 \right\| < \beta$, we drop the constraint corresponding to λ_j , giving $\underline{G}_{m-1}^T \underline{p}_{k+1} = \underline{0}$ and $(\underline{g}^m)^T \underline{p}_{k+1} < 0$ as required. Most of the time it will also be true that $(\underline{g}^j)^T \underline{p}_{k+1} \leq 0$ $j = m+1 \cdots m+r$, but in a degenerate case we will have $(\underline{g}^j)^T \underline{p}_{k+1} > 0$ for some j of the

linearly dependent set. (Note that this can occur only if one of the set relied on $\underline{g}^m$ for its linear dependence on $\underline{g}_m$).

The procedure in a degenerate case is to replace the b_j 's in equation (4.2a) by

$$b_{m+1} + \xi, \dots, b_{m+r} + r\xi \quad \xi > 0$$

Now only the independent hyperplanes intersect at $\underline{x}_k$. The $m+r$ hyperplanes form an unbounded convex region R , with the point $\underline{x}_k$ a vertex of R and $\underline{g}_k^0$ a constant vector. We now solve the sub-problem :

$$\text{minimize } (\underline{g}_k^0)^T \underline{x} \text{ over } R$$

This problem will have one of two possible solutions :

- (i) $(\underline{g}_k^0)^T \underline{x}$ remains bounded on R . That is, the solution point is a vertex of R . Since for $\xi \rightarrow 0$ all vertices coincide with $\underline{x}_k$, we cannot improve the value of $(\underline{g}_k^0)^T \underline{x}$ and therefore conclude we are at the overall minimum.
- (ii) There exists an optimal ray $\underline{x} = \underline{x}_k + \alpha_k \underline{p}_{k+1}$ (with $\alpha_k \geq 0$ and $\underline{x}_k$ a vertex of R) which is in R pointing out the open end, along which the linear function $(\underline{g}_k^0)^T \underline{x}$ assumes arbitrarily low values. In this case

$$(\underline{p}_{k+1})^T \underline{g}_k^0 < 0 \text{ and } (\underline{p}_{k+1})^T \underline{g}^j \leq 0 \quad j = 1 \dots m+r$$

so $\underline{p}_{k+1}$ is a feasible direction along which the algorithm is restarted.

Note that this sub-problem can be solved by any linear-programming method.

Degeneracy seems to be mainly of theoretical interest. The same sort of degeneracy can arise in linear programming by the simplex method. While it is possible to construct examples where a degenerate situation does arise, it has not been known to arise in practical problems and

indeed most commercial linear programming codes do not have degeneracy procedures incorporated in them⁽³²⁾.

4.5.4 Rounding Errors, x-Correction

Rounding errors are liable to build up in the recursive updating of the inverse moment matrix as constraints continually get added to and dropped from the basis. As a result, the current point, $\underline{x}_k$, may have moved off the surface of a constraint in the current basis. The procedure is as follows :

(i) Calculate $f^j(\underline{x}_k)$ $j = 1 \text{ --- } m+r$. This includes a possible linearly dependent subset of r constraints.

(ii) If $|f^j(\underline{x}_k)|$ $j = 1 \text{ --- } m+r$ is outside a certain tolerance, ξ_2 , then an x-correction is required :

$$\underline{x}_k^1 = \underline{x}_k - \underline{S}_k \underline{G}_m (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} (\underline{G}_m^T \underline{x}_k - \underline{b})$$

We can only consider m constraints for this correction, usually the constraints in the basis. But there is no reason why the vector $\underline{b}$ should not include some of the b_j 's from the linearly dependent subset. For we have :

$$\begin{aligned} f^{m+r}(\underline{x}_k) &= (\underline{G}^{m+r})^T \underline{x}_k - b_{m+r} \\ &= \underline{\alpha}^T \underline{G}_m^T \underline{x}_k - b_{m+r} \end{aligned}$$

$$(\text{since } \underline{G}^{m+r} = \underline{G}_m \underline{\alpha} \text{ for some } \underline{\alpha}).$$

$$\begin{aligned} \text{Hence, } f^{m+r}(\underline{x}_k^1) &= \underline{\alpha}^T \underline{G}_m^T \underline{x}_k^1 - b_{m+r} \\ &= \underline{\alpha}^T \underline{G}_m^T \underline{x}_k - b_{m+r} - \underline{\alpha}^T (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_k \underline{G}_m) (\underline{G}_m^T \underline{x}_k - b_{m+r}) \\ &= 0 \text{ as required.} \end{aligned}$$

Other checks for rounding errors are :

(iii) The projection at a vertex should be zero (within a certain error tolerance).

(iv) If dropping a constraint is expected to increase the norm of a projection, and it does not, rounding errors are evident.

In this case the inverse Hessian approximation is set to the identity, and the moment matrix reformed and inverted.

4.5.5 Initial Feasible Point and Starting Procedure

Since the initial approximation to the inverse Hessian is usually the identity matrix, the initial projection operator is analogous to Rosen's. Hence, the generation of an initial feasible point and the starting procedure are analogous to his procedures, which are briefly described below.

Initial Feasible Point :

(i) Add all the strict equalities to the initial basis (testing for linear dependence).

(ii) If one or more constraints in the basis are violated, an $\underline{x}$ -correction is required (section 4). The basis constraints are then satisfied.

(iii) If any constraint not in the basis is violated, the one most violated is tested for linear independence and, if the test is satisfied, added to the basis. The procedure is then repeated from step (ii).

(iv) If this constraint is linearly dependent, test for a constraint to drop. If there is not one to drop, there is no feasible $\underline{x}$.

(v) If there is a constraint to drop, drop the constraint and repeat from (ii).

Starting Procedure : Given an initial feasible point.

(i) Set $p_1 = -g_0^0$

(ii) Add strict equalities to basis, and from the direction vector

$$p_1 = -P_{m,0} S_0 g_0^0$$

(iii) If j^* inequality constraints are present, calculate

$$\omega = \max_j p_1^T g^j \quad j = 1 \text{ --- } j^*$$

(iv) If $\omega \leq 0$, add the corresponding constraint to the basis (testing for linear dependence). Set $m = m+1$. Form the new direction vector $p_1 = -P_{m,0} S_0 g_0^0$, decrease j^* by 1 and repeat from step (iii).

4.5.6 Numerical Examples

Programmes using both Methods A and B with the projection operator were tested on four examples. One of these was a simple quadratic, used to test the properties of the methods, and the other 3 are to be found in Colville's study of nonlinear programming codes⁽³³⁾. This is the only set of test problems to be found in the literature dealing with a comprehensive range of characteristics. The details of the examples are tabulated in Appendix 2, as also is the progress of each method on them.

Example 1

This was a simple example - a 4-dimensional quadratic function, to be minimized subject to 7 inequality constraints. Both methods took 5 steps and 6 function evaluations to find the minimum. For quadratics, both methods will have generated the inverse Hessian of the quadratic function

after n linearly independent steps. But convergence may be achieved in more or less steps than $n+1$ depending on when one reaches the correct subspace. After n linearly independent steps all subsequent steps will be linearly dependent on these, but in fact nothing gets added to the Hessian inverse and subsequent steps are always to the minimum of the current subspace.

Example 2

This example consists of a five-dimensional cubic function, to be minimized subject to 15 inequality constraints. Method A took 9 steps and 10 function evaluations, and Method B took 6 steps and 7 function evaluations. Both compare well with the results of Goldfarb and Lapidus, who report 9 steps and 13 function evaluations with their method and 12 steps, 29 function evaluations for Rosen's Gradient Projection method.

Example 3

This is a 16-variable quartic, subject to 8 linear equality constraints and 32 inequality constraints in the form of upper and lower bounds on the variables. The initial starting point was infeasible, approximately one third of the total computing time was spent in finding an initial feasible starting point. Method A took 11 steps and 19 function evaluations, and Method B took 5 steps and 9 function evaluations. This compares with 6 steps, 10 function evaluations with the Goldfarb-Lapidus method and 7 steps 14 function evaluations with Rosen's Gradient Projection. Although Method A took over twice the number of function evaluations of Method B, the increase in computing time was only 20%. This is due to the computational advantage it has of

not having to use recursion formulae repeatedly; however this is offset by the disadvantage that we are suppressing information by storing the constraint normals $\underline{g}^j$ in the columns of $\underline{Q}_k$.

Example 4

This is the same four-dimensional banana as example 4 in section (3.6), except upper and lower bounds of ± 10.0 are set on the variables. These are usually not active, but they decreased the number of function evaluations in Method B from 68 to 43 (35 steps). This decrease is due to the fact that the bounds limit the initial step size, (also, one variable was constrained at its bounding positive value for two early steps.) Method A took 66 steps, 206 function evaluations. Goldfarb and Lapidus report 105 steps, 142 function evaluations with their method, and Rosen's Gradient Projection failed completely, being unable to reduce the function below 7.87.

The conclusion drawn from these results is that, despite the need to use recursion formulae to update the inverse moment matrix, Method B is far superior to Method A. It would also appear to be more powerful than other current methods of minimizing subject to linear constraints.

4.6 A Strategy for Nonlinear Constraints

4.6.1 Introduction

In section 4.4 the relationship between the projection operator and the Lagrangian function was discussed. It is possible to take this discussion further in deriving an algorithm for handling nonlinear as well as linear constraints.

In the general case any search path will give rise to points away from the constraint boundaries, so that $\underline{f}_{k-1} \neq 0$ and the step $\underline{p}_k$ derived in section 4.1 takes the more general form (c.f. (4.9) and (4.11)):

$$\underline{p}_k = -\alpha_{k-1} \underline{S}_{k-1} (\underline{E}_{k-1}^0 - \underline{G}_m \underline{\lambda}_{m,k-1}) \quad (4.60)$$

$$\text{where } \underline{\lambda}_{m,k-1} = (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \left[\underline{G}_m^T \underline{S}_{k-1} \underline{E}_{k-1}^0 - \underline{f}_{k-1} \right] \quad (4.61)$$

Comparing the expression (4.60) with the step for the unconstrained case given by (3.22) it can be seen that (4.60) constitutes an unconstrained step on the Lagrangian function (with the Lagrange Multipliers taken as constant), whose gradient is given by :

$$\underline{E}_{k-1}^L = \underline{E}_{k-1}^0 - \underline{G}_m \underline{\lambda}_{m,k-1} \quad (4.62)$$

Now the Hessian of the Lagrangian function

$$\underline{H}^L = (\underline{H}^0 - \sum_{j=1}^m \lambda_j \underline{H}^j) \quad (4.63)$$

is not equal to $\underline{H}^0$ when nonlinear constraints are present, since the constraint Hessians, $\underline{H}^j$, are not all zero. It would thus seem a reasonable proposition to note the differences in $\underline{E}_{k-1}^j$, which form the columns of $\underline{G}_m$, and update the approximating matrix, $\underline{S}_{k-1}$, using differences in $\underline{E}_{k-1}^L$. $\underline{S}_k$ then becomes an approximation to the inverse of $\underline{H}^L$.

We are thus using Lagrange multipliers to put constraint nonlinearities into the objective function and approximating a solution to :

$$\underline{g}_{k-1}^0 + (\underline{H}^0 - \sum_{j=1}^m \lambda_j \underline{H}^j)(\underline{\bar{x}} - \underline{x}_{k-1}) = \underline{g}_m \underline{\lambda} \quad (4.64)$$

$$\text{subject to } \underline{f}_{k-1}^j + (\underline{g}_{k-1}^j)^T(\underline{\bar{x}} - \underline{x}_{k-1}) = 0 \quad j = 1 \text{ --- } m \quad (4.65)$$

This approach has some similarities to that of Wilson⁽²³⁾, discussed in section 2.2.3; however, there are essential differences in that he forms (and stores) each Hessian matrix separately, and solves a sequence of quadratic programming problems formed by truncating the objective and constraint functions at the second-order term of the Taylor's series expansions.

4.6.2 Inequality Constraints

It is evident that a major problem arises in the case of nonlinear inequality constraints, since once a constraint, say j , is dropped from the basis there is no way of dropping the contribution of the $\lambda_j \underline{H}^j$ term in the approximation to the inverse of $\underline{H}^L$. Unless one projects back onto the nonlinear constraint surface, there is also a combinatorial problem in deciding which set of constraints to include in the basis, since the linearized constraint equations are displaced from the current point unless this point itself satisfies the constraints.

$$\text{i.e. the surface } \underline{G}_m^T(\underline{x} - \underline{x}_k) + \underline{f}_k = 0$$

does not pass through $\underline{x}_k$ unless $\underline{f}_k = 0$.

One way of overcoming these difficulties is to turn all nonlinear constraints into equality constraints by the use of slack variables.

There are two possible formulations :

$$\begin{aligned}
 1. \quad & \text{Replace} \quad f^j(\underline{x}) \leq 0 \\
 & \text{by} \quad f^j(\underline{x}) + x_{n+j} = 0 \\
 & \quad \quad \quad x_{n+j} \geq 0
 \end{aligned} \tag{4.66}$$

$$\begin{aligned}
 2. \quad & \text{Replace} \quad f^j(\underline{x}) \leq 0 \\
 & \text{by} \quad f^j(\underline{x}) + x_{n+j}^2 = 0
 \end{aligned} \tag{4.67}$$

The first formulation has the disadvantage that it replaces the nonlinear inequality constraint by a nonlinear equality constraint plus a linear inequality, whereas the second formulation provides only a nonlinear equality. However, this disadvantage is not serious since the constrained step strategy of section 4.5 handles linear inequalities easily. Moreover, the search area is restricted to half the space of the second formulation. The first formulation is also helpful computationally in that when the linear constraint on x_{n+j} is active this indicates that the original nonlinear inequality is active as an equality constraint - this fact will be used to some advantage in section 4.6.4.

Of course computational experience is the final basis on which to judge the relative merits of each formulation; however, there is one case where the first is the obvious choice, and that is to handle nonlinear inequalities of the form :

$$a_j \leq f^j(\underline{x}) \leq b_j$$

This is replaced by only one nonlinear equality constraint and two linear inequalities :

$$\begin{aligned}
 f^j(\underline{x}) - x_{n+j} &= 0 \\
 a_j &\leq x_{n+j} \leq b_j
 \end{aligned} \tag{4.68}$$

The idea of using slack variables to handle inequality constraints certainly removes all of the theoretical difficulties involved in formulating an algorithm, however each slack variable adds another dimension to the problem and it is not hard to envisage instances where the problem rapidly assumes enormous proportions. In the implementation of such an approach some device would be required to overcome this difficulty. For example, it is probably not worth considering a constraint until it is actually violated, and then treat it as an equality with its associated slack variable set at its bounding value. (Note that the problem of the $\lambda_j \underline{H}^j$ term contaminating the inverse approximation only arises on dropping a constraint, not on adding one).

4.6.3 Penalty Terms

As noted in section 4.4, $\underline{H}^L = \underline{H}^0$ if the constraints are linear so that if $\underline{H}^0$ is positive-definite the stationary point of the Lagrangian will be a minimum and correspond to the constrained minimum of the original objective function. In the presence of nonlinear constraints $\underline{H}^L = \underline{H}^0 - \sum_{j=1}^m \lambda_j \underline{H}^j$ may not be positive-definite; indeed if slack variables are added as given by (4.66) it will be singular. There is thus no guarantee that the stationary point of the Lagrangian will correspond to the constrained minimum of the original objective function.

Arrow and Solow⁽¹⁸⁾ have proposed an algorithm based on a modified Lagrangian, of the form :

$$L(\underline{x}, \underline{\lambda}) = f^0(\underline{x}) - \sum_{j=1}^m \lambda_j f^j(\underline{x}) + \frac{1}{2}K \sum_{j=1}^m f^j(\underline{x})^2 \quad (4.69)$$

They show that for sufficiently large positive K the Hessian of this Lagrangian is positive-definite. This formulation is equivalent in our

case to adding a penalty term to the objective function, so that it becomes :

$$f^o(\underline{x}) = \overline{f}^o(\underline{x}) + \frac{1}{2}K \sum_{j=1}^m f^j(\underline{x})^2 \quad (4.70)$$

where $f^j(\underline{x}) = 0$, $j = 1 \dots m$, are nonlinear equality constraints, and $\overline{f}^o(\underline{x})$ is the original objective function.

Considering only nonlinear constraints for the present, we note however that :

$$\underline{E}^o(\underline{x}) = \overline{\underline{E}}^o(\underline{x}) + \underline{G}_m(K \underline{f}(\underline{x})) \quad (4.71)$$

where $\underline{f}(\underline{x}) = f^j(\underline{x}) \quad j = 1 \dots m$.

Hence the contribution of the penalty term to the predicted step gets cancelled out :

$$\begin{aligned} \text{from (4.61)} : \quad \underline{\lambda}_{m,k-1} &= (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \left[\underline{G}_m^T \underline{S}_{k-1} (\overline{\underline{E}}_{k-1}^o - \underline{G}_m(K \underline{f}_{k-1})) - \underline{f}_{k-1} \right] \\ &= (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_{k-1} \overline{\underline{E}}_{k-1}^o - \underline{f}_{k-1}) - K \underline{f}_{k-1} \end{aligned}$$

and from

$$\begin{aligned} (4.60) : \quad \underline{p}_k &= -\alpha_{k-1} \underline{S}_{k-1} ((\overline{\underline{E}}_{k-1}^o - \underline{G}_m(K \underline{f}_{k-1})) - \underline{G}_m \underline{\lambda}_{m,k-1}) \\ &= -\alpha_{k-1} \underline{S}_{k-1} (\overline{\underline{E}}_{k-1}^o - \underline{G}_m (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_{k-1} \overline{\underline{E}}_{k-1}^o - \underline{f}_{k-1})) \end{aligned}$$

which is the same as if the penalty term was not present. However, the presence of the penalty term does become important when determining the value of the step-size parameter α_{k-1} . Indeed the prime usefulness of the penalty term is to give a measure of progress towards the solution, since the modified objective function can be made to decrease at each step. By the same token it prevents large violations of any constraint;

however, a very large value of K in the penalty term would give rise to very small steps.

4.6.4 Projection for Feasibility

In Rosen's⁽⁷⁾ extension of his gradient projection algorithm to include nonlinear constraints, the current point is projected orthogonally back onto the constraint surface at the end of each step. The same device may be used with a variable-metric projection operator, the correction term being :

$$\underline{x}_k = \underline{x}'_k - \underline{S}_{k-1} \underline{G}_m (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \underline{f}'_k \quad (4.72)$$

Iteration is needed for nonlinear constraints, since each corrected $\underline{x}_k$ satisfies only the linearized constraint equations.

Preliminary experiments showed that this device was useful, and indeed necessary for the final convergence to the solution. The reason for this is evident from the discussion in section 4.6.3 - we wish to keep K in the penalty term low to prevent small step lengths, yet unless K is sufficiently high the value of the modified objective function is not a good monitor and the process converges to a point outside the feasible region with an objective function value lower than that at the true constrained minimum. This is precisely the situation which gives rise to the sequential penalty function approach discussed in section 1.3 and is one which we wish to avoid. It may well be that such a point is sufficiently close to the true minimum that a quadratic approximation is valid and we merely have to look for a stationary point of the Lagrangian without noting function values, but again in order to guarantee this the choice of K becomes critical.

In order to prevent unnecessary implementation of this step for nonlinear inequality constraints, a rule adopted was to force feasibility on a constraint only when the constraint on its associated slack variable was active, i.e. when the original inequality constraint was active.

4.6.5 Algorithm

Formulate the problem in the form :

$$\text{minimize } f^0(\underline{x}) = \bar{f}^0(\underline{x}) + \frac{1}{2K} \sum_{j=1}^{n_L} f^j(\underline{x})^2 \quad (4.73)$$

subject to

$$\text{nonlinear constraints } f^j(\underline{x}) = 0 \quad j = 1 \text{ --- } n_L \quad (4.74)$$

$$\text{linear constraints } (\underline{g}^j)^T \underline{x} - b_j \leq 0 \quad j = n_L + 1, \text{ --- } m^* \quad (4.75)$$

Note that the nonlinear constraints (4.74) include nonlinear inequalities which have been transformed into equalities using slack variables, and (4.75) includes possible bounds on the slack variables. (A register of those constraints in (4.75) which correspond to the slack variables in (4.74) is also kept). Linear equality constraints are included in (4.75), the only special condition on them being that they are never considered for dropping from the basis.

The presence of nonlinear constraints in the basis does not affect the treatment of linear constraints, which are handled in exactly the same way as described in section 4.5.

Let $m \geq n^u$ be the number of constraints in the basis at the current point. A general step of the algorithm is as follows :

1. Project :

$$(i) \text{ Form } \underline{\lambda}_{m,k-1} = (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \left[\underline{G}_m^T \underline{S}_{k-1} \underline{g}_{k-1}^0 - \underline{f}_{k-1} \right] \quad (4.61)$$

$$\text{where } \underline{f}_{k-1} = \begin{cases} f^j(\underline{x}_{k-1}) & j = 1 \text{ --- } n^u \\ 0 & j = n^u + 1, \text{ --- } m. \end{cases}$$

The rules for adding and dropping linear constraints carry over from section 4.5.

$$(ii) \text{ If } \underline{\lambda}_{m,k-1} \leq \underline{\epsilon} \text{ and } \left\| \underline{S}_k(\underline{g}_k^0 - \underline{G}_m \underline{\lambda}_{m,k-1}) \right\| \leq \underline{\epsilon} \text{ stop;}$$

the Kuhn-Tucker conditions for an overall minimum are satisfied.

2. Step :

$$(i) \text{ Calculate } \alpha'_{k-1} = \min_j ((\underline{g}^j)^T \underline{x}_{k-1} - b_j) / (\underline{g}^j)^T \underline{S}_{k-1} (\underline{g}_{k-1}^0 - \underline{G}_m \underline{\lambda})$$

$$j = m+1, \text{ --- } m^x$$

$$= \text{distance to nearest constraint.}$$

$$(ii) \text{ Step } \underline{x}'_k - \underline{x}_{k-1} = - \alpha_{k-1} \underline{S}_{k-1} (\underline{g}_{k-1}^0 - \underline{G}_m \underline{\lambda}_{m,k-1})$$

$$\text{where } \alpha_{k-1} = \min(1.0, \alpha'_{k-1})$$

(iii) Adjust α_{k-1} if necessary by cubic interpolation to satisfy :

$$f^0(\underline{x}_{k-1}) - f^0(\underline{x}'_k) \geq \delta \cdot |\alpha_{k-1}| \cdot \left| (\underline{g}_{k-1}^0)^T \underline{S}_{k-1} (\underline{g}_{k-1}^0 - \underline{G}_m \underline{\lambda}_{m,k-1}) \right|$$

3. Project for Feasibility :

If any nonlinear constraints are deemed to be active (as discussed

in section 4.6.4), step :

$$\underline{x}_k = \underline{x}_k' - \underline{S}_{k-1} \underline{G} (\underline{G}^T \underline{S}_{k-1} \underline{G})^{-1} \underline{f}_k' \quad (4.72)$$

where

$$\underline{f}_k' = \begin{cases} f^j(\underline{x}_k') & j = 1 \text{ --- } n^l \\ 0 & j = n^l + 1, \text{ --- } m \end{cases}$$

Repeat this step if necessary until $|f^j(\underline{x}_k)| \leq \epsilon$ for the active constraints.

4. Update :

$$(i) \text{ Form } \underline{g}_k^L = (\underline{E}_k^0 - \underline{E}_{k-1}^0) - \sum_{j=1}^{n^l} \lambda_{m,k-1}^j (\underline{E}_k^j - \underline{E}_{k-1}^j) \quad (4.76)$$

Note that $\lambda_{m,k-1}$ is not updated for this step.

(ii) Update $\underline{S}_{k-1}$ to $\underline{S}_k$ using (3.17),

where $\underline{p}_k = \underline{x}_k - \underline{x}_{k-1}$ for the complete step (2 and 3)

$$\text{and } \underline{g}_k = \underline{g}_k^L.$$

Note that for the test for positive-definiteness, $\underline{z}_k^T \underline{S}_{k-1}^{-1} \underline{p}_k / c_k > 0$,

we have :

$$\underline{S}_{k-1}^{-1} \underline{p}_k = -\alpha_{k-1} (\underline{E}_{k-1}^0 - \underline{G}_m \lambda_{m,k-1}) - \underline{G}_m (\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1} \left(\sum_{i=1}^r \underline{f}_i \right)$$

where $\underline{f}_i$ is the value of $\underline{f}_k'$ at the i^{th} iteration, $i = 1 \text{ --- } r$, of step 3.

If the test for positive-definiteness fails, update using

$$c_k = \underline{z}_k^T \underline{z}_k, \text{ as previously.}$$

(iii) Update $(\underline{G}_m^T \underline{S}_{k-1} \underline{G}_m)^{-1}$ to $(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1}$ using (4.24).

(iv) Drop $\underline{E}_{k-1}^j$, $j = 1 \text{ --- } n^l$, from $(\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1}$ using (4.36), and replace these with $\underline{E}_k^j$, $j = 1 \text{ --- } n^l$, using (4.29).

Return to step 1.

If there is a large number of nonlinear constraints compared to linear constraints present, it is probably not worth updating the inverse moment matrix by recursion, but easier to solve a set of linear equations for $\lambda_{m,k}$ in step 1.

4.6.6 Numerical Examples

The programme for linear constraints was modified to include the features described above for handling nonlinear constraints, and tested on a number of examples. Appendix 3 contains details of these examples, together with tables showing the progress of solution on each.

Example 1 This was a simple problem consisting of a 3-dimensional quadratic objective function, subject to one quadratic equality constraint and 6 linear inequality constraints. This converged in 4 steps, 9 function evaluations and 8 nonlinear constraint evaluations.

Example 2 This was one of Colville's⁽³³⁾ test problems. It consisted of a 5-dimensional quadratic objective function, subject to 6 quadratic inequality constraints and 10 linear inequality constraints. The nonlinear inequality constraints consisted of upper and lower bounds on 3 quadratic functions. This problem was transformed by use of slack variables, (using equation (4.68)), to an 8-dimensional problem with 3 quadratic equality constraints and 16 linear inequality constraints. This problem took 5 steps, 9 function evaluations and 9 nonlinear constraint evaluations. This result compares well with figures quoted in Colville's report for other algorithms.

Example 3 This was perhaps the most difficult problem in Colville's set of standard examples; a large number of algorithms appearing in his report failed to converge to the solution. It is the dual problem to Example 2 in Section 3.6, and consists of a 15-dimensional cubic objective function, subject to 5 quadratic inequality constraints and 15 linear inequalities. This problem was also transformed by the use of slack variables, (using equation (4.66)), into a 20-dimensional problem with 5 quadratic equality constraints and 20 linear inequality constraints.

This problem took 56 steps, 178 function evaluations and 119 nonlinear constraint evaluations. This result is comparable to figures quoted by Davies⁽⁹⁾ with his method for nonlinear constraints. Both show a significant improvement on penalty function methods quoted in Colville's report.

CHAPTER 5

CONVERGENCE THEOREMS5.1 Finite Convergence for Quadratic Functions5.1.1 Minimization without ConstraintsTheorem 1

Suppose we have a set of steps, $\underline{p}_i$, $1 \leq i \leq k$, not necessarily consecutive, which satisfy the equation :

$$\underline{S}_k \underline{g}_i = \underline{p}_i \quad 1 \leq i \leq k \quad (5.1)$$

with $\underline{S}_k$ non-singular

Then, for a quadratic function, a step according to :

$$\underline{p}_{k+1} = - \underline{S}_k \underline{g}_k \quad (5.2)$$

is either linearly independent of the steps, $\underline{p}_i$, or else attains a stationary point of the function.

Proof Suppose that $\underline{p}_{k+1}$ is linearly dependent on the $\underline{p}_i$:

$$\underline{p}_{k+1} = \sum_{i=1}^k \beta_i \underline{p}_i \quad (5.3)$$

where at least one β_i is non-zero.

Since the function is quadratic :

$$\underline{g}_i = \underline{H} \underline{p}_i \quad (5.4)$$

From (5.3) and (5.4) :

$$\underline{g}_{k+1} = \underline{H} \underline{p}_{k+1} = \underline{H} \sum_{i=1}^k \beta_i \underline{p}_i = \sum_{i=1}^k \beta_i \underline{g}_i \quad (5.5)$$

From (5.1), (5.3) and (4.5) :

$$p_{k+1} = \sum_{i=1}^k \beta_i p_i = \sum_{i=1}^k \beta_i \underline{S}_k q_i = \underline{S}_k q_{k+1} \quad (5.6)$$

From (5.2) and (5.6), since $\underline{S}_k$ is non-singular :

$$-E_k = q_{k+1} = E_{k+1} - E_k$$

whence :

$$E_{k+1} = 0$$

This is the condition for attaining a stationary point, which is thus a necessary condition for linear dependence of p_{k+1} on the set p_i , $1 \leq i \leq k$.

Corollary 1.1

Q.E.D.

The stationary point is attained in at most $(n+1)$ steps. This result follows immediately from the fact that the maximum possible number of linearly independent steps is n .

Although step (5.2) is always the first estimate for p_{k+1} , the proposed methods subsequently modify this by the factor $\alpha_k \neq 1$ if the function increases. However, this does not change the direction of the step, which therefore remains linearly independent of the set of p_i . The theorem therefore shows that we still build up a set of linearly independent p_i step by step.

No information is rejected for Method B, so the corollary is immediately applicable to this method; however, this statement must be qualified by an assumption of stability, i.e. that $\underline{S}_k$ remains finite and positive definite.

For Method A, however, we have to show that no vector $\underline{q}_i$ is subsequently eliminated in updating $\underline{R}_k$ and $\underline{S}_k$. To show that $\underline{q}_{k+1}$ does not eliminate any previous $\underline{q}_i$ we need to assume that $\underline{H}$ is non-singular. Then if $\underline{q}_{k+1}$ is linearly dependent on the set $\underline{q}_i$:

$$\underline{q}_{k+1} = \sum_{i=1}^k \beta_i \underline{q}_i \quad (5.7)$$

We have using (5.4) :

$$\underline{p}_{k+1} = \underline{H}^{-1} \underline{q}_{k+1} = \sum_{i=1}^k \beta_i \underline{H}^{-1} \underline{q}_i = \sum_{i=1}^k \beta_i \underline{p}_i \quad (5.8)$$

Again this leads to the conclusion that the attainment of a stationary point is a necessary condition for linear dependence of $\underline{q}_{k+1}$ on the set $\underline{q}_i$, $1 \leq i \leq k$. If we do not step to a stationary point, $\underline{q}_{k+1}$ is linearly independent of the previous $\underline{q}_i$, and since $\underline{Q}_k$ is non-singular there must therefore be a non-zero element $\underline{e}_j^T \underline{R}_k \underline{q}_{k+1}$ among the rows which do not correspond to this set.

We must also show that $j \neq m$ if we need to include $\underline{g}_k$ among the columns of $\underline{Q}_{k+1}$ using equations (3.32) applied to $\underline{R}_{k+1}$. Now suppose that $\underline{g}_k$ is linearly dependent on the set $\underline{q}_i$, $1 \leq i \leq k+1$:

$$\underline{g}_k = - \sum_{i=1}^{k+1} \beta_i \underline{q}_i \quad (5.9)$$

Then from (5.1), (5.2) and (5.9) :

$$\underline{p}_{k+1} = - \underline{S}_k \underline{g}_k = \sum_{i=1}^{k+1} \beta_i \underline{S}_k \underline{q}_i = \sum_{i=1}^k \beta_i \underline{p}_i + \beta_{k+1} \underline{S}_k \underline{q}_{k+1} \quad \dots (5.10)$$

But from (5.1) and (3.14) :

$$\underline{p}_{k+1} = \underline{S}_{k+1} \underline{q}_{k+1} = \underline{S}_k \underline{q}_{k+1} + (\underline{p}_{k+1} - \underline{S}_k \underline{q}_{k+1}) \cdot \underline{y}_{k+1}^T \underline{q}_{k+1} \quad (5.11)$$

Equation (5.11) gives a linear relationship between $\underline{p}_{k+1}$ and $\underline{S}_k \underline{q}_{k+1}$, so equation (5.10) implies that $\underline{p}_{k+1}$ is linearly dependent on the previous $\underline{p}_i$, $1 \leq i \leq k$. Once again this leads to the conclusion that attainment of a stationary point is a necessary condition for $\underline{g}_k$ to be linearly dependent on the set $\underline{q}_i$, $1 \leq i \leq k+1$, and if we do not step to a stationary point there will be a non-zero element $\underline{e}_m^T \underline{R}_{k+1} \underline{g}_k$ among the rows of $\underline{R}_{k+1}$ which do not correspond to this set of $\underline{q}_i$. It then follows from Corollary 1.1 that we attain the minimum in at most $(n+1)$ steps.

5.1.2 Minimization with Linear Constraints

Theorem 2

Suppose we have a set of steps $\underline{p}_i$, $1 \leq i \leq k$, not necessarily consecutive, which satisfy equation (5.1) with $\underline{S}_k$ non-singular. Then for a quadratic function, $f^0(\underline{x})$, a step according to :

$$\underline{p}_{k+1} = \underline{S}_k \underline{G}_m (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_k \underline{g}_k^0 - \underline{f}_k) - \underline{S}_k \underline{g}_k^0 \quad (5.12)$$

is either linearly independent of the steps, $\underline{p}_i$, or else attains a stationary point in the space formed by the intersection of the constraint set :

$$\underline{G}_m^T (\underline{x} - \underline{x}_k) + \underline{f}_k = 0$$

Proof As in Theorem 1 we suppose that $\underline{p}_{k+1}$ is linearly dependent on the $\underline{p}_i$, so that equations (5.3) - (5.6) hold. From (5.12) and (5.6), since

$\underline{S}_k$ is non-singular :

$$\underline{G}_m (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_k \underline{E}_k^0 - \underline{f}_k) - \underline{E}_k^0 = \underline{q}_{k+1} = \underline{E}_{k+1}^0 - \underline{E}_k^0 \quad (5.13)$$

Thus :

$$\underline{E}_{k+1}^0 = \underline{G}_m (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_k \underline{E}_k^0 - \underline{f}_k) = \underline{G}_m \underline{\lambda} \quad (5.14)$$

$$\text{where } \underline{\lambda} = (\underline{G}_m^T \underline{S}_k \underline{G}_m)^{-1} (\underline{G}_m^T \underline{S}_k \underline{E}_k^0 - \underline{f}_k)$$

Equation (5.14) shows that $\underline{E}_{k+1}^0$ is orthogonal to the intersection of the constraint set : $\underline{G}_m^T (\underline{x} - \underline{x}_k) + \underline{f}_k = 0$, which is the condition for a stationary point in this intersection.

Hence the attainment of a stationary point in the intersection of the constraint set is a necessary condition for linear dependence of $\underline{p}_{k+1}$ on the set $\underline{p}_i$, $1 \leq i \leq k$. Q.E.D.

Corollary 2.1

The stationary point in the intersection of m linear constraints is attained in at most $(n - m + 1)$ steps in this intersection.

Corollary 2.2

If the intersection of a set of linear constraints has been spanned by previous steps, then a step according to (5.12) attains immediately a stationary point in this intersection.

Again the modification of (5.12) by a factor $\alpha_k \neq 1$ if the step is not successful does not change the direction of the step and we build up a set of linearly independent $\underline{p}_i$ within the intersection of the constraints step by step.

The properties of the two methods carry over from the unconstrained case, so that Corollaries 2.1 and 2.2 apply to them, subject in the case of Method A to non-singularity of the Hessian matrix.

5.2 Overall Convergence

Although a non-singular Hessian matrix was implicit in the basis of the methods presented, we have already seen that this assumption is unnecessary for minimization of quadratic functions, and it is also possible to prove ultimate convergence to a stationary point for a more general class of functions. The proofs are applicable to both methods developed here and are analogous to a proof given by Ostrowski⁽³⁴⁾ for a similar class of descent methods. However, the conditions used here have the advantage that they may be computationally verified.

5.2.1 Minimization without Constraints

Theorem 3

Suppose that $f(\underline{x})$ is defined on $U \subset E^n$ and is such that :

- (i) $\Omega' = \{ \underline{x} \mid \underline{x} \in U, f(\underline{x}) \leq C \}$ is a bounded subset of U .
- (ii) $f(\underline{x})$ has continuous second derivatives on Ω' and there is a $\wedge$ such that :

$$\|H(\underline{x})\| \leq \wedge, \quad \underline{x} \in \Omega' \quad (5.15)$$

We take a point $\underline{x}_0 \in \Omega' = \{ \underline{x} \mid \underline{x} \in U, f(\underline{x}) \leq C \}$ with $g(\underline{x}_0) \neq 0$ and generate a sequence $\underline{x}_0, \underline{x}_1, \dots, \underline{x}_k, \underline{x}_{k+1}, \dots$ according to the rule :

$$\underline{p}_{k+1} = \underline{x}_{k+1} - \underline{x}_k = -\alpha_k \underline{S}_k \underline{g}_k \quad (5.16)$$

with $\underline{S}_k$ chosen to satisfy the conditions :

$$|\underline{g}_k^T \underline{S}_k \underline{g}_k| \geq \delta \cdot \|\underline{g}_k\| \cdot \|\underline{S}_k \underline{g}_k\| \quad (5.17)$$

$$c \|\underline{g}_k\| \leq \|\underline{S}_k \underline{g}_k\| \leq \sigma \|\underline{g}_k\| \quad (5.18)$$

where δ, c and σ are fixed positive constants.

Then we can always choose a finite non-zero α_k at each step such that :

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) \geq \left\{ \alpha_k \underline{g}_k^T \underline{S}_k \underline{g}_k \right\} > 0 \quad (5.19)$$

with ϵ a fixed positive number less than unity.

With the α_k so chosen, the sequence $\underline{x}_k$ lies in Ω and tends to $\Omega^* = \{ \underline{x} \mid \underline{x} \in \Omega, \underline{g}(\underline{x}) = 0 \}$ in the sense that the distance of $\underline{x}_k$ from Ω^* tends to zero as $k \rightarrow \infty$.

Proof If $\underline{x}_k$ is in Ω then since Ω is an open subset, neighbouring points are also in Ω and hence for sufficiently small $\underline{p}_{k+1}$ both $\underline{x}_{k+1}$ and $\underline{x}_k + \theta \underline{p}_{k+1}$ are in Ω and we can expand $f(\underline{x}_{k+1})$ in a Taylor series about $\underline{x}_k$, with remainder of second order :

$$f(\underline{x}_{k+1}) = f(\underline{x}_k) + \underline{g}_k^T \underline{p}_{k+1} + \frac{1}{2} \underline{p}_{k+1}^T \underline{H}(\underline{x}_k + \theta \underline{p}_{k+1}) \underline{p}_{k+1} \quad 0 \leq \theta \leq 1 \quad \dots (5.20)$$

We choose the sign of α_k to make $\alpha_k \underline{g}_k^T \underline{S}_k \underline{g}_k$ positive as required by (5.19) and substitute (5.16) into (5.20) to give :

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) = \alpha_k \underline{g}_k^T \underline{S}_k \underline{g}_k \left\{ 1 - \frac{1}{2} \frac{|\alpha_k| (\underline{S}_k \underline{g}_k)^T \underline{H}(\underline{x}_k + \theta \underline{p}_{k+1}) (\underline{S}_k \underline{g}_k)}{|\underline{g}_k^T \underline{S}_k \underline{g}_k|} \right\} \quad \dots (5.21)$$

We can use (5.15), (5.17) and (5.18) in (5.21) to give :

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) \geq \alpha_k \underline{g}_k^T \underline{S}_k \underline{g}_k \left\{ 1 - \frac{1}{2} \frac{\lambda_5}{\delta} \cdot |\alpha_k| \right\} \quad (5.22)$$

Thus so long as :

$$|\alpha_k| \leq \frac{2\delta(1-\epsilon)}{\delta-\epsilon} \quad (5.23)$$

$\underline{x}_{k+1}$ remains in Ω and (5.19) is satisfied. Since $\underline{x}_0 \in \Omega$ the whole sequence is in Ω and it is therefore always possible to choose a finite non-zero α_k to satisfy condition (5.19).

With α_k so chosen, the sequence $f(\underline{x}_k)$ is monotonically decreasing, and since (ii) implies that $f(\underline{x})$ is bounded below in Ω , it is also convergent, so that :

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) \rightarrow 0 \text{ as } k \rightarrow \infty \quad (5.24)$$

But from (5.17), (5.18) and (5.19) :

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) \geq \epsilon \cdot \delta \cdot \epsilon \cdot \|\underline{g}_k\|^2 \cdot |\alpha_k| \quad (5.25)$$

and since $\alpha_k \neq 0$ (5.24) implies that $\|\underline{g}_k\| \rightarrow 0$ and hence $\underline{g}_k \rightarrow 0$.

Further, from (5.16) and (5.18) :

$$\|\underline{p}_{k+1}\| = |\alpha_k| \cdot \|\underline{g}_k\| \leq \epsilon |\alpha_k| \cdot \|\underline{g}_k\| \quad (5.26)$$

and it follows that $\underline{p}_{k+1} \rightarrow 0$ as $k \rightarrow \infty$.

Now suppose that the sequence does not tend to Ω^* . Then since the sequence $\underline{x}_k$ is in Ω' , a closed bounded region of R^n , we can find a subsequence, $\underline{x}_i$ for which

$$|\underline{x}_i, \Omega^*| \rightarrow d > 0$$

But this subsequence in turn contains a subsequence converging to a point, $\underline{z}$ say, contained in Ω , so that

$$|\underline{z}, \Omega^*| = d > 0 \quad (5.27)$$

But $g_k \rightarrow 0$, so that we have $g(\underline{z}) = 0$ and $\underline{z}$ belongs to Ω^* . This contradicts (5.27), so that the sequence $\underline{x}_k$ must tend to Ω^* , and the proof is complete.

We note further that if the basic procedure converges to a stationary point which is not a (weak) local minimum, then the local search procedure will restart the basic procedure at a point with a lower function value, which in view of (5.19) excludes convergence to the same point again. We shall therefore ultimately converge to a (weak) local minimum.

5.2.2 Minimization with Linear Constraints

If the minimization is subject to a set of linear equality constraints on $\underline{x}$, it will be easy to find an initial feasible point, and we can then make use of the following theorem on convergence of the projection algorithm, which is closely analogous to Theorem 3 :

Theorem 4

Given the set $S = \{ \underline{x} \mid \underline{x} \in \mathbb{E}^n, \underline{G}_m^T \underline{x} = \underline{b} \}$ and the function $f^0(\underline{x})$ defined on S such that :

- (i) $\Omega' = \{ \underline{x} \mid \underline{x} \in S, f^0(\underline{x}) \leq c \}$ is a bounded sub-set of S .
- (ii) $f^0(\underline{x})$ has continuous second derivatives on Ω and there is a λ such that :

$$\| \underline{H}^0(\underline{x}) \| \leq \lambda, \quad \underline{x} \in \Omega \quad (5.28)$$

We take a point $\underline{x}_0 \in \Omega = \{ \underline{x} \mid \underline{x} \in S, f(\underline{x}) < c \}$ with $\underline{g}^0(\underline{x}_0) \notin S^\perp$ and generate a sequence $\underline{x}_0, \underline{x}_1, \dots, \underline{x}_k, \underline{x}_{k+1}, \dots$ according to the rule :

$$\underline{p}_{k+1} = \underline{x}_{k+1} - \underline{x}_k = -\alpha_k \underline{s}_k \underline{g}_k^L \quad (5.29)$$

$$\text{where } \underline{E}_k^L = \underline{E}_k^O - \underline{G}_m \underline{\lambda}_{m,k} \quad (5.30)$$

$$\underline{\lambda}_{m,k} \text{ satisfies : } \underline{G}_m^T \underline{S}_k \underline{G}_m \underline{\lambda}_{m,k} = \underline{G}_m^T \underline{S}_k \underline{E}_k^O \quad (5.31)$$

and the $\underline{S}_k$ are chosen to satisfy the conditions for the solution of (5.31) to exist, and :

$$\left| (\underline{E}_k^L)^T \underline{S}_k (\underline{E}_k^L) \right| \geq \delta \left\| \underline{E}_k^L \right\|^2 \quad (5.32)$$

$$\left\| \underline{E}_k \underline{E}_k^L \right\| \leq \sigma \left\| \underline{E}_k^L \right\| \quad (5.33)$$

where δ and σ are fixed positive constants.

Then we can always choose a finite non-zero α_k at each step such that :

$$f^O(\underline{x}_k) - f^O(\underline{x}_{k+1}) \geq \xi \alpha_k (\underline{E}_k^L)^T \underline{S}_k \underline{E}_k^L > 0 \quad (5.34)$$

with ξ a fixed positive constant less than unity.

With the α_k so chosen, the sequence $\underline{x}_k$ lies in Ω and tends to $\Omega^\infty = \{ \underline{x} | \underline{x} \in \Omega, \underline{g}(\underline{x}) \in S^\perp \}$ in the sense that the distance of $\underline{x}_k$ from Ω^∞ tends to zero as $k \rightarrow \infty$.

Proof From (5.30) and (5.31) :

$$\underline{G}_m^T \underline{S}_k \underline{E}_k^L = \underline{G}_m^T \underline{S}_k \underline{E}_k^O - \underline{G}_m^T \underline{S}_k \underline{G}_m \underline{\lambda}_{m,k} = 0 \quad (5.35)$$

$$\text{whence } (\underline{E}_k^L)^T \underline{S}_k \underline{E}_k^L = (\underline{E}_k^O)^T \underline{S}_k \underline{E}_k^L + \underline{\lambda}_{m,k}^T \underline{G}_m^T \underline{S}_k \underline{E}_k^L = (\underline{E}_k^O)^T \underline{S}_k \underline{E}_k^L \quad (5.36)$$

Since $\underline{x}_0 \in S$, it follows from (5.29) and (5.35) that all points of the sequence $\underline{x}_k$ lie in S . For sufficiently small α_k , $\underline{x}_{k+1}$ therefore lies in Ω and we may expand $f^O(\underline{x}_{k+1})$ in a Taylor series about $\underline{x}_k$, with a remainder of second order :

/please see over

$$f^0(\underline{x}_{k+1}) = f^0(\underline{x}_k) + (\underline{g}_k^0)^T \underline{p}_{k+1} + \frac{1}{2} \underline{p}_{k+1}^T \underline{H}^0(\underline{x}_k + \theta \underline{p}_{k+1}) \underline{p}_{k+1} \quad (5.37)$$

$$0 \leq \theta \leq 1$$

We choose the sign of α_k to make $\alpha_k (\underline{g}_k^L)^T \underline{S}_k \underline{g}_k^L$ positive, substitute (5.29) into (5.37) and use (5.36) to give :

$$f^0(\underline{x}_k) - f^0(\underline{x}_{k+1}) = \alpha_k (\underline{g}_k^L)^T \underline{S}_k \underline{g}_k^L \left\{ 1 - \frac{1}{2} \frac{|\alpha_k| (\underline{S}_k \underline{g}_k^L)^T \underline{H}^0(\underline{x}_k + \theta \underline{p}_{k+1}) (\underline{S}_k \underline{g}_k^L)}{(\underline{g}_k^L)^T \underline{S}_k \underline{g}_k^L} \right\} \quad \dots (5.38)$$

Now if $\underline{x}_{k+1}$ lies in Ω , so also does $(\underline{x}_k + \theta \underline{p}_{k+1})$, so we may substitute (5.28), (5.32) and (5.33) into (5.38) to give :

$$f^0(\underline{x}_k) - f^0(\underline{x}_{k+1}) \geq \alpha_k (\underline{g}_k^L)^T \underline{S}_k \underline{g}_k^L \left\{ 1 - \frac{1}{2} \frac{\underline{\lambda}_k^2}{\delta} |\alpha_k| \right\} \quad (5.39)$$

Thus $\underline{x}_{k+1}$ remains in Ω , and (5.34) is satisfied, so long as :

$$|\alpha_k| \leq \frac{2\delta(1-\xi)}{\underline{\lambda}_k^2} \quad (5.40)$$

It is therefore always possible to choose a finite non-zero α_k at each step to satisfy condition (5.34).

With the α_k so chosen, the sequence $f(\underline{x}_k)$ is monotonically decreasing, and since (5.28) implies that $f(\underline{x})$ is bounded below on S , it is also convergent, so that

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) \rightarrow 0 \text{ as } k \rightarrow \infty \quad (5.41)$$

But from (5.32) and (5.34) :

$$f(\underline{x}_k) - f(\underline{x}_{k+1}) \geq \xi \cdot \delta \cdot |\alpha_k| \|\underline{g}_k^L\|^2$$

and since $|\alpha_k| \neq 0$ (5.41) implies that $\|\underline{g}_k^L\| \rightarrow 0$ and hence $\underline{g}_k^L \rightarrow 0$.

Further, from (5.29) and (5.33) :

$$\|p_{k+1}\| = |\alpha_k| \left\| \frac{s_k}{\|s_k\|} \frac{L}{\|L\|} \right\| \leq \sigma \left\| \frac{L}{\|L\|} \right\|^2$$

and it follows that :

$$p_{k+1} \rightarrow 0 \text{ as } k \rightarrow \infty$$

It can also be shown as before that :

$$\left\| \frac{x_k}{\|x_k\|}, \frac{L}{\|L\|} \right\| \rightarrow 0 \text{ as } k \rightarrow \infty.$$

If the minimization is subject to a set of linear inequality constraints (with or without linear equality constraints) we are still assured of convergence if we satisfy the appropriate conditions at each step. This follows from the fact that the arguments in Theorems 3 and 4 show that we can choose a finite non-zero α_k at each step and remain in the region Ω appropriate to the set of active constraints at that step. Since each Ω is bounded, their union is also bounded and we have sufficient conditions for convergence of the sequence $f^0(x_k)$; the remainder of the proof follows as before.

Frequently the feasible region $V = \{x | x \in E^n; \frac{G}{m}^T x = \frac{b}{m}; \frac{G}{r}^T x \leq \frac{b}{r}\}$ is bounded, and provided that condition (ii) of the theorems is satisfied for all $x \in V$ we shall not then require condition (i) on the set Ω in either theorem. In fact we can replace Ω in Theorem 3 by the interior of V , and in Theorem 4 Ω may be replaced by the interior of $V \cap S$. This follows from the fact that the search strategy and projection algorithm ensure that all points of the sequence x_k are feasible and hence lie in a bounded region of E^n .

PART III

OPTIMIZATION IN
CHEMICAL ENGINEERING DESIGN

CHAPTER 6

PROGRAMME FOR THE CALCULATION OF A
MULTICOMPONENT PLATE DISTILLATION COLUMN

6.1 Introduction

This chapter describes a computer programme to calculate the steady-state performance of a multicomponent plate distillation column. It is a 'performance' rather than a 'design' programme in that the input variables are those for the simulation of an existing column. However, such a programme can be used as part of a 'function calculation' in some design or optimization problem.

A paper describing the method of calculation used in the programme has been published⁽³⁵⁾. A summary of this is given, including a discussion of the various convergence techniques tried.

6.2 The Describing Equations

An overall mass balance around plate p gives the equation :

$$F_p + E_{p-1} + V_{p-1} + L_{p+1} = (V_p + P_p^V) + (L_p + E_p + P_p^L) \quad (6.1)$$

where F_p = molar rate of feed to plate p .

E_p = molar rate of liquid entrainment leaving plate p .

V_p = molar rate of vapour leaving plate p .

P_p^V = molar rate of vapour product leaving plate p .

L_p = molar rate of liquid leaving plate p .

P_p^L = molar rate of liquid product leaving plate p .

The mass balance for a typical component, i , similarly yields :

$$\begin{aligned} f_{i,p} + E_{p-1} x_{i,p-1} + V_{p-1} y_{i,p-1} + L_{p+1} x_{i,p+1} = & (V_p + P_p^V) y_{i,p} + \\ & + (L_p + E_p + P_p^L) x_{i,p} \\ & \dots (6.2) \end{aligned}$$

where $x_{i,p}$ = mole fraction of component i in the liquid leaving plate p .

$y_{i,p}$ = mole fraction of component i in the vapour leaving plate p .

$f_{i,p}$ = molar rate of feed of component i to plate p .

The energy balance gives the equation :

$$\begin{aligned} Q_p + h_{fp} F_p + h_{p-1} E_{p-1} + H_{p-1} V_{p-1} + h_{p+1} L_{p+1} = & H_p (V_p + P_p^V) + \\ & + h_p (L_p + E_p + P_p^L) \quad (6.3) \end{aligned}$$

where Q_p = Indirect heat input to plate p.

h_{fp} = specific molar enthalpy of feed to plate p.

h_p = specific molar enthalpy of liquid leaving plate p.

H_p = specific molar enthalpy of vapour leaving plate p.

Equations (6.1), (6.2) and (6.3) apply equally well to the reboiler (stage 1) and the condenser (stage P), if we make the following special substitutions :

$$\text{For the reboiler : } V_0 = E_0 = L_1 = 0 \quad (6.4)$$

$$\text{For the condenser : } V_P = E_P = L_{P+1} = 0 \quad (6.5)$$

The Murphree vapour-phase plate efficiency relates the actual change in vapour composition with that achieved by equilibrium. We define this as :

$$e_{i,p}^v = \frac{y_{i,p} - y_{i,p-1}}{y_{i,p}^* - y_{i,p-1}} \quad (6.6)$$

where $y_{i,p}^*$ is the mole fraction of component i in the vapour which would be in equilibrium with the liquid leaving the stage if this were brought to its boiling point ($\theta_{i,p}^*$) at the pressure in question. The equilibrium relationship is of the form :

$$y_{i,p}^* = K_{i,p} x_{i,p} \quad (6.7)$$

where $y_{i,p}^*$ is the mole fraction of component i in the vapour in equilibrium with the liquid at its boiling point at a given pressure, and $K_{i,p}$ is the equilibrium ratio of component i, a function of temperature and pressure for ideal mixtures, and also of composition for non-ideal mixtures. It follows from this definition that the equilibrium ratios, $K_{i,p}$, for each component i, are calculated at the boiling point

of the liquid mixture corresponding to $x_{i,p}$.

The thermal efficiency relates actual changes in vapour and liquid temperatures with those achieved by thermal equilibrium between the mixtures :

$$\text{For the liquid : } \phi_p^L = \frac{\theta_{L,p} - \theta_{L,p+1}}{\theta_{L,p}^* - \theta_{L,p+1}} \quad (6.8)$$

$$\text{For the vapour : } \phi_p^V = \frac{\theta_{v,p} - \theta_{v,p-1}}{\theta_{v,p}^* - \theta_{v,p-1}} \quad (6.9)$$

$$\text{or } \phi_p^V = \frac{\theta_{v,p} - \theta_{v,p-1}}{\theta_{L,p}^* - \theta_{v,p-1}} \quad (6.10)$$

where $\theta_{L,p}$ = temperature of liquid leaving plate p.

$\theta_{L,p}^*$ = boiling point of liquid leaving plate p.

$\theta_{v,p}$ = temperature of vapour leaving plate p.

$\theta_{v,p}^*$ = dew point of vapour leaving plate p.

Difficulty with the reboiler and condenser is most easily avoided by treating these as ideal stages, however special performance relations may be applied.

The component mass-balance equations (6.2) have a band structure if we work with vapour mole fractions. Using (6.7) to express $y_{i,p}^*$ in terms of the $y_{i,p}$ and subsequently using (6.6) to express $x_{i,p}$ in terms of the $y_{i,p}$, we obtain a set of equations of the form :

$$g_{p,p-2}^i y_{i,p-2} + g_{p,p-1}^i y_{i,p-1} + g_{p,p}^i y_{i,p} + g_{p,p+1}^i y_{i,p+1} = f_{i,p} \quad (6.11)$$

$$\text{or in matrix form : } \underline{G}^i \underline{y}_i = \underline{f}_i \quad (6.12)$$

where $\underline{y}_i$ and $\underline{f}_i$ are column vectors with y_{ip} and f_{ip} as p^{th} elements respectively, and $\underline{G}^i$ is a band matrix whose non-zero elements are :

$$\left. \begin{aligned} g_{p,p-2}^i &= (1 - \eta_{i,p-1}^v) \frac{E_{p-1}}{K_{i,p-1} \eta_{i,p-1}^v} \\ g_{p,p-1}^i &= -V_{p-1} - \frac{E_{p-1}}{K_{i,p-1} \eta_{i,p-1}^v} - (1 - \eta_{i,p}^v) \frac{L_p + P_p^L}{K_{i,p} \eta_{i,p}^v} \\ g_{p,p}^i &= (V_p + P_p^v) + \frac{E_p + L_p + P_p^L}{K_{i,p} \eta_{i,p}^v} + (1 - \eta_{i,p}^v) \frac{L_{p+1}}{K_{i,p+1} \eta_{i,p+1}^v} \\ g_{p,p+1}^i &= - \frac{L_{p+1}}{K_{i,p+1} \eta_{i,p+1}^v} \end{aligned} \right\} (6.13)$$

From the special conditions for the condenser and reboiler, we have :

$$g_{1,-1}^i = g_{1,0}^i = g_{P,P+1}^i = 0$$

If there are C components in the mixture, there will be C matrices of the form of (6.12). The equations are of course nonlinear, for the coefficients of $\underline{G}^i$ $i = 1 \text{ --- } C$, are functions of composition. For a trial set of coefficients, the $\underline{y}_i$ will not usually sum to unity and iterative adjustment of the coefficients is required. The sum of the individual mass balances will satisfy the overall mass balance (6.1) when the individual compositions sum to unity, but we must also satisfy the overall energy balance equations (6.3), which form a set of P nonlinear equations. However, for a trial set of enthalpies of the vapour and liquid streams, these may be solved for the L_p and V_p using (6.1) with (6.3) by successive substitution.

The variables specified for the performance calculation are as follows :

- a) The total number of plates, P , and the positions of all feeds and sidestreams.
- b) Flow-rates, composition, and heat content of each feed.
- c) State and flow-rate of each product (except one flow-rate which is calculated by the overall column mass balance).
- d) Indirect heat input for each plate (except one which is calculated by the overall column energy balance).
- e) All plate pressures.

6.3 Solution of the Band Matrix

For a trial set of coefficients (6.13) we have a set of C matrices of the form (6.12) which must be solved for the vapour composition. These matrices are often ill-conditioned, and to reduce rounding errors, the method of solution proposed by Wilkinson⁽³⁶⁾ was used. This consists of triangular decomposition with partial pivoting - a variant of the Gaussian elimination procedure where the matrix is decomposed into the product of a lower and an upper triangular matrix. These are subsequently solved by back-substitution. The usual process is to work from the top diagonal element downwards, rewriting the current row and column as a product of the lower and upper triangular elements already formed. Interchanges with remaining rows of the matrix are made to ensure that the current diagonal element is of maximum modulus.

Since there is one super-diagonal band and two sub-diagonal bands, it is evident that in order to preserve this band structure as much as possible and thereby save storage space, it is better to work backwards from the bottom diagonal element. Thus, there is a possibility of pivoting only with the superdiagonal element, so a row interchange will not alter the band structure much. In fact, only $7 \times n$ storage locations are needed, including a $1 \times n$ register of pivot operations.

Iterative refinement of the solution is also included. This involves repeated back-substitution with the triangular matrices on the residuals until they become sufficiently small.

6.4 Convergence Techniques

Iterative adjustment of the coefficients of (6.12) is required until the calculated vapour compositions on each plate sum to unity, and the overall plate mass and energy balances are all satisfied simultaneously. The method of solution is to decompose this adjustment into two iterative loops. In the inner loop, the equilibrium ratios $K_{i,p}$, and possibly the Murphree plate efficiencies $\eta_{i,p}^v$, are adjusted iteratively until the vapour compositions sum to unity (within a prescribed error tolerance). Each time this inner loop has converged the internal flowrates are recalculated from the plate mass and energy balances, using the enthalpies corresponding to the current compositions. This forms an outer iterative cycle, which must be repeated until there is no change in these variables between two successive cycles.

(a) Inner Loop

It is reasonable to expect that when $\sum_{i=1}^c y_{i,p} = 1$ for each plate p , then also $\sum_{i=1}^c y_{i,p}^* = 1$ and $\sum_{i=1}^c x_{i,p} = 1$. Indeed this must be so unless the $\eta_{i,p}^v$ are not the same for each component on a given plate, whence this condition is one solution. Since the equilibrium ratios, $K_{i,p}$, given by equation (6.6) in conjunction with (6.7), correspond to the boiling point of a liquid composition of $x_{i,p}$ $i = 1 \text{ --- } c$, or to the dew point of a vapour composition of $y_{i,p}^*$ $i = 1 \text{ --- } c$, we may adjust $K_{i,p}$ in the inner iterative loop by seeking to satisfy the relation

$$\sum_{i=1}^c y_{i,p}^* / K_{i,p} = 1 \quad p = 1 \text{ --- } P \quad (6.14)$$

Three convergence techniques were tried, as follows :

Method 1 : Newton's Method on the function :

$$R(\theta) = \sum_{i=1}^c y_{i,p}^* / K_{i,p} - 1.0 \quad (6.15)$$

we thus make a step in temperature :

$$(\theta)_{n+1} = (\theta)_n - \frac{(R)_n}{(dR/d\theta)_n} \quad (6.16)$$

for each plate p , and recalculate $K_{i,p}$ $i = 1 \text{ --- } c$ using this new estimate of $\theta_{L,p}^*$. Of course, this is feasible only for ideal mixtures, where $K_{i,p}$ is a function of temperature only for a given pressure.

Method 2 : Second-order Newton-Raphson to minimize the function :

$$S(\theta) = \left(\sum_{i=1}^c y_{i,p}^* / \alpha_{i,j,p} - K_{j,p} \right)^2 \quad (6.17)$$

where $\alpha_{i,j,p} = K_{i,p} / K_{j,p}$, the relative volatility of component i to component j , the base component.

We thus make a step in temperature :

$$(\theta)_{n+1} = (\theta)_n - \frac{(dS/d\theta)_n}{(d^2S/d\theta^2)_n} \quad (6.18)$$

for each plate p , and again recalculate $K_{i,p}$ $i = 1 \text{ --- } c$ using this new estimate of $\theta_{L,p}^*$. Again, the mixture must be ideal. Note also from (6.17) that if the $\alpha_{i,j,p}$ were constant, then we would effectively be considering only $K_{j,p}$ by this step in temperature and changing the other $K_{i,p}$ for plate p by the same factor throughout.

Method 3 : Relaxation.

This is based on the assumption of constant relative volatility $\alpha_{i,j,p}$ for each plate p . The appropriate correction derived from (6.14) is then :

$$(K_{j,p})_{\text{corr.}} = (K_{j,p})_n \cdot \left(\sum_{i=1}^c y_{i,p}^* / K_{i,p} \right)_n \quad (6.19)$$

Again, using the assumption of constant relative volatility, (6.19) amounts to scaling all the $K_{i,p}$ for a given plate by a constant factor σ_p .

$$(K_{i,p})_{\text{corr.}} = (K_{i,p})_n \cdot \sigma_p \quad i = 1 \text{ --- } c \quad (6.20)$$

$$\text{where } \sigma_p = \left(\sum_{i=1}^c y_{i,p}^* / K_{i,p} \right)_n \quad (6.21)$$

Rather than use the corrected $K_{i,p}$ directly, convergence is improved by using over-relaxation according to the formula :

$$(K_{i,p})_{n+1} = \omega (K_{i,p})_{\text{corr.}} + (1 - \omega)(K_{i,p})_n \quad (6.22)$$

where ω is a relaxation parameter, which may be given a fixed value, or may be based on a linear extrapolation of previous values. (Note that the $K_{i,p}$ are recalculated with the boiling points in the outer loop.)

None of the methods converged satisfactorily unless the compositions, $y_{i,p}^*$, were scaled so that $\sum_{i=1}^c y_{i,p}^* = 1$. On the test problem used to develop the programme, Method 1 and Method 3 were divergent and Method 2 was convergent but very slow. However, all three methods converged well when the compositions were scaled. Method 2 was slightly better than Method 1 on a number of examples tried, but this was later abandoned as it was considered that there was little justification for the calculation of second-derivatives which, unless equilibrium ratios are expressed as

polynomials in temperature, is an arduous task. Method 3 is by far the most useful, since it is not dependent on the form of the equilibrium data.

(b) Outer Loop

The new compositions calculated in the inner loop give new values for the enthalpies of the vapour and liquid mixtures on each plate. A plate to plate energy balance, using (6.3) together with (6.1), will thus give new values for the internal flowrates.

Recalculating these variables from the new compositions amounts to direct substitution, and it is evident that convergence of this outer loop may be accelerated by the use of a variable relaxation parameter, as in the inner loop. This device proved sufficiently successful to justify the increased storage requirement.

CHAPTER 7

AN OPTIMIZATION PROBLEM IN DISTILLATION COLUMN DESIGN7.1 Introduction

The programme described in Chapter 6 is for the simulation of an existing column. In the design of a column, some of the variables used as input to this programme must be adjusted to meet other design specifications.

The feed composition, enthalpy, and flowrate are usually fixed by previous units in the process. For a conventional column, with one feedplate and two product streams, the variables which may be adjusted are the number of plates, feedplate position, one of the product flowrates and one of the external heat loads, either the reboiler rate or condenser rate.

If the design specifications are a product rate and composition, the problem is usually approached in one of two ways : either the number of plates is fixed and the reboiler rate varied to meet the desired composition specification, or else the reboiler rate is fixed and the number of plates adjusted to meet the composition specification. In both cases the product rate must also be varied if the design specifications are given as two compositions.

In this chapter we consider the problem of meeting product composition specifications for a column with a fixed number of plates. This is more convenient for using the programme described in Chapter 6; and also of course the number of plates is an integer variable whereas the reboiler rate is a continuous variable.

For a given feedplate position, the reboiler rate (or condenser rate) and the top product flowrate must be varied in order to meet two product composition specifications.

This may be done by formulating an unconstrained minimization problem, where these two variables are adjusted to minimize the sum of squares of the deviations of the product compositions from specification. (There are bounds on the variables, in that they must both be greater than zero, and also the product rate must be less than the total feedrate, but these are never active in any physically meaningful situation.)

Another design problem is to determine the optimal feedplate position. For a medium-sized column, consisting of 20 theoretical plates or less with one feedplate and two product streams, this problem is easily solved by trial and error since the choice of feedplate positions is not large. The simplest course of action is to determine the reboiler rate and product flowrate to meet the composition specifications for each feedplate position, and determine which position minimizes the operating cost.

For a column with a large number of plates and possibly multiple feedstreams, the problem is not easily solved and rightly belongs to the field of Mixed-Integer Programming. However, an interesting problem may be formulated by allowing feed to enter on any plate for a predetermined range of plates. For each such plate the feed flowrate becomes another variable. An interesting set of linear constraints arise from this formulation : each feed flowrate must be greater than or equal to zero, and also there is a linear equality constraint in that the sum of the individual flowrates must equal the predetermined total feed flowrate.

7.2 Product Specifications for a Single Feedplate

A specific example was taken in order to try out the above ideas in practice. In order to keep the programme running time within the express turn-around limitations of the computing system, a necessarily small distillation column was used as an example. It consisted of a 15 plate column, distilling a 4-component hydrocarbon mixture at a pressure of 300 psia. Feed of the composition given below entered at a rate of 100 moles/hr., and an enthalpy of 15,700 Btu/mole.

$C_3 H_8$	0.15
i - C_4	0.25
n - C_4	0.30
n - C_5	0.30

Enthalpy and equilibrium data were taken from Holland⁽³⁷⁾.

Plate performance characteristics were specified as follows :

Entrainment Ratio	$R_p = E_p/V_p = 10\%$	$p = 1 \dots P-1$
Murphree Plate Efficiency	$\eta_{i,p}^v = 60\%$	$i = 1 \dots c, p = 2 \dots P-1$
Thermal Efficiency of Liquid		
Phase	$\phi_p^L = 90\%$	$p = 2 \dots P-1$
Thermal Efficiency of Vapour		
Phase	$\phi_p^V = 90\%$	$p = 2 \dots P-1$

The reboiler and condenser were treated as ideal stages.

The product specifications were set as follows :

S1 : The recovery of n - C_4 in the bottoms must be 95%

S2 : The mole fraction of $C_3 H_8$ in the distillate must be 0.50.

The first problem discussed in section 7.1 was solved for each of four feedplate positions, for feed entering on plates 7, 8, 9 and 10, adjusting the condenser rate and the top product flowrate. The minimization method used was Algorithm B.3a with Reset 2, using Stewart's⁽³¹⁾ algorithm to calculate gradients numerically, as discussed in section 3.7.

For feed entering on the 7th plate, the progress of solution is shown in Table 7.1 below :

TABLE 7.1
Feed on 7th Plate

Step	No. Col. Calcns.	Pp moles/hr.	- Qp Btu/hr.	$\frac{\Delta S_1}{S_1} \%$	$\frac{\Delta S_2}{S_2} \%$
0	1	40.000	$1.000 \cdot 10^6$	- 27.450	- 56.000
1	10	26.252	$9.734 \cdot 10^6$	- 2.660	- 21.640
2	15	22.232	$9.622 \cdot 10^6$	+ 0.670	- 1.388
3	19	22.159	$9.620 \cdot 10^6$	+ 0.708	- 0.996
4	22	22.103	$9.618 \cdot 10^6$	+ 0.750	- 0.626

The problem was resolved for feed entering on plates 8, 9 and 10. For both plates 8 and 9 the starting point was the minimum point for the previous plate, while for plate 10 the starting point was that for plate 7.

The results are shown in Table 7.2 below :

TABLE 7.2
Final Values for each Plate Position

Feedplate	Pp	- Qp	$\frac{\Delta S_1}{S_1} \%$	$\frac{\Delta S_2}{S_2} \%$
7	22.103	$9.618 \cdot 10^6$	+ 0.750	- 0.626
8	21.937	$4.002 \cdot 10^6$	+ 0.437	- 0.350
9	21.817	$3.086 \cdot 10^6$	+ 0.188	- 0.045
10	22.058	$5.940 \cdot 10^6$	+ 0.402	- 0.574

7.3 Product Specifications for Multiple Feedplates

The other problem discussed in section 7.1, that of allowing the feed to distribute itself over a predetermined range of plates, was tried on this system also.

The two product composition specifications given in section 7.2 may be cast as nonlinear equality constraints, and the objective function to be minimized is the operating cost (the cost of reboiler heat and condenser cooling water). This problem thus uses the methods discussed in Part 2 to their fullest extent, as it not only includes linear and nonlinear constraints, but also requires numerical evaluation of derivatives.

The reboiler heat cost was specified at 10^{-6} cost units/Btu, and the cost of the condenser cooling water at 10^{-7} cost units/Btu. Using these unit costs, the cost obtained from each feedplate position discussed in section 7.2 is given in Table 7.3 below :

TABLE 7.3
Profit for Single Feedplate
(Basis: 100 mole/hr.feed)

Plate Position	Reboiler Cost	Condenser Cost	Total Cost
7	9.7025	0.9618	10.6643
8	4.0852	0.4002	4.4854
9	3.1679	0.3086	3.4765
10	6.0235	0.5940	6.6175

The problem discussed above was solved, using as a starting point all the feed onto plate 9. The composition specifications of section 7.2 were required to be met within 0.75% tolerance.

$$\text{i.e.} \quad \left| \frac{\Delta S_j}{S_j} \right| \% \leq 0.75 \quad j = 1, 2.$$

The progress of solution is shown in Table 7.4 below :

TABLE 7.4

Product Specifications for Multiple Feedplates

No. Col. Calcns.	Feed 7 mole/hr.	Feed 8 mole/hr.	Feed 9 mole/hr.	Feed 10 mole/hr.	P _p ^v mole/hr.	-Q _p x10 ⁶ Btu/hr.	$\frac{\Delta S_1}{S_1}$ % ¹	$\frac{\Delta S_2}{S_2}$ % ²	Total Cost
1	0.0,0	0.0,0	100.000	0.0,0	21.817	3.086	+0.188	-0.045	3.4765
16	0.14,-5	0.50,-5	99.498	0.65,-6	21.483	1.535	0.189	-0.243	1.7682
23	-0.25,-2	-0.46,-2	99.494	-0.46,-2	21.245	0.833	-0.426	-1.172	0.992
25	-0.13,-1	-0.23,-2	99.496	-0.23,-2	21.364	1.183	-0.267	-0.470	1.378

Note : The extra digits denote the exponent at 10. The negative values are within the error tolerance imposed on the constraints.

7.4 DISCUSSION OF RESULTS

The performance of the minimization method on this design problem was not impressive. This poor performance, particularly on the formulation discussed in section 7.3, is due partly to the fact that the example distillation column was very small and the feed distribution over the range of plates considered made very little difference. This of course gave rise to small gradients of the objective function with respect to the associated variables, and indeed there was little change in these variables from their initial values.

The problem of small gradients would have an aggravating effect on the most serious problem arising in these examples, namely that the numerical calculation of gradients was inaccurate. Blame for this should not be laid entirely at Stewarts' algorithm, even though the problems discussed in section 3.7 showed poor convergence. Results could have been considerably improved if a systematic study of the correct scaling of the variables had been made. However, the need for such a study in itself demonstrates the weakness of the algorithm.

However, the methods do give a clear indication that the optimal feedplate position is plate 9. Also, there is no improvement in operating cost to be obtained by spreading the feed over a range of plates. The cost decrease shown in Table 7.4 was obtained by reducing the reboiler rate and condenser rate, at the expense of a decrease in top product flow-rate and a decrease in the accuracy to which the composition specifications were satisfied.

GENERAL CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK

It may be concluded from the numerical experiments that the methods presented are extremely powerful gradient methods of minimization. Implementation of the algorithms requires a moderate amount of matrix-vector manipulation; they are therefore particularly suited to problems where calculation of the objective function is by far the most expensive step computationally. There is also a need to store at least one full matrix (although advantage may be taken of symmetry in the matrices discussed). Storage limitations of the computer, and the possibility of rounding error, thus limit the size of the problems which may be conveniently handled.

For unconstrained minimization, Method B (Algorithm 3a) would seem to be rather better than the Fletcher-Powell method, and the results of the numerical experiments certainly establish it as an attractive alternative.

For minimization subject to linear constraints, the properties of the method developed here remove all the theoretical difficulties associated with the method of Goldfarb and Lapidus, and numerical tests show a consistent improvement upon their results.

The extensions of the algorithm to deal with both linear and nonlinear constraints have produced encouraging results. No proofs of convergence in the presence of nonlinear constraints have been produced, but satisfactory proofs have been given for the case of linear constraints and for the unconstrained case. Other slack variable formulations to deal with nonlinear inequality constraints could be tried, although this is no major undertaking.

The area in which further work is definitely required is in the numerical evaluation of derivatives. The methods presented here fall short of the required objective, namely to have an algorithm capable of being incorporated into a general-purpose simulation and design programme, and in order to achieve this the methods must be extended to deal with problems where gradients are not available explicitly.

REFERENCES

1. POLLARD G.P. and SARGENT R.W.H., "Off Line Computation of Optimum Controls for a Plate Distillation Column", paper presented at I.F.A.C. Congress, Warsaw, (June 1969).
2. HESTENES M.R. and STEIFEL E., "Methods of Conjugate Gradients for Solving Linear Systems", J. Res. N.B.S. 49, 409 (1952).
3. FLETCHER R. and REEVES C.M., "Function Minimization by Conjugate Gradients", The Computer Journal, 7, 149 (1964).
4. MYERS G.E., "Properties of the Conjugate Gradient and Davidon Methods", J. Optimzn. Theory and Applns. 2(4), 209 (1968).
5. DAVIDON W.C., "Variable Metric Method for Minimization", A.E.C., Research and Development Report ANL-5990 (Rev.)(1959).
6. FLETCHER R. and POWELL M.J.D., "A Rapidly Convergent Descent Method for Minimization", The Computer Journal 6, 163 (1963).
7. ROSEN J.B., "The Gradient Projection Method for Nonlinear Programming", J. Soc. Ind. Appl. Math. 8, 181 (1960) and 9, 514 (1961).
8. GOLDFARB D. and LAPIDUS L., "A Conjugate Gradient Method for Nonlinear Programming", paper presented at A.I.Ch.E. 61st. National Meeting, Houston (February, 1967).
9. DAVIES D., "The Use of Davidon's Method in Nonlinear Programming", I.C.I. Management Services Report MSDH/68/110 (August, 1968).
10. DANTZIG G.B., "Linear Programming and Extensions", (Princeton University Press, 1963).
11. GRIFFITHS R.E. and STEWART R.A., "A Nonlinear Programming Technique for the Optimization of Continuous Processing Systems", Management Science 7, 379 (1961).
12. KELLEY Jr. J.E., "The Cutting-Plane Method for Solving Convex Programmes", J. Soc. Ind. Appl. Math. 8, 703 (1960).

REFERENCES (continued)

13. COURANT R., "Variational Methods for the Solution of Problems of Equilibrium and Vibrations", Bull. Am. Math. Soc. (January 1943).
14. FIACCO A.V. and McCORMICK G.P., "The Sequential Unconstrained Minimization Technique for Nonlinear Programming; A Primal-Dual Method", Management Science 10, 360 (1964).
15. COURANT R. and HILBERT D., "Methods of Mathematical Physics", Vol. 1, p. 231 (Interscience, New York, 1962).
16. KUHN H.W. and TUCKER A.W., "Nonlinear Programming", Proc. Second Berkeley Symposium, p. 481 (University of California Press, Berkeley, 1951).
17. BERGE C and GHOUILA-HOURI A., "Programming, Games, and Transportation Networks", p. 78 (Methuen, London, 1965).
18. ARROW K.J., HURWICZ L. and UZAWA H. (Eds.), "Studies in Linear and Nonlinear Programming", (Stanford University Press, 1958).
19. KING R.P., "Necessary and Sufficient Conditions for Inequality Constrained Extreme Values", Ind. Eng. Chem. (Fund.) 5, 484 (1966).
20. TAKAHASHI I., "Variable Separation Principle for Mathematical Programming", J. Op. Res. Soc. Japan 6 (2), 82 (1964).
21. FALK J.E., "Lagrange Multipliers and Nonlinear Programming", J. Math. Anal. and Applns. 19, 141 (1967).
22. BROSILOW C. and LASDON L., "A Two Level Optimization Technique for Recycle Processes", A.I.Ch.E. - I.Chem.E. Symposium Series No. 4, p. 75 (Instn. Chem. Engrs., London 1965).
23. WILSON R., "A Simplicial Algorithm for Concave Programming" discussed by E.M.L. Beale in "Nonlinear Programming", J. Abadie ed., p. 192 (North Holland, 1967).
24. BODEWIG E., "Matrix Calculus", p. 32 (North Holland, Amsterdam 1956).

REFERENCES (continued)

25. AITKEN A.C., "Determinants and Matrices", p. 87, (Oliver and Boyd, London 1939).
26. BROYDEN C.G., "Quasi-Newton Methods and their Application to Function Minimization" Math. Comp. 21, 368 (July, 1967).
27. DAVIDON W.C., "Variance Algorithm for Minimization", The Computer Journal, 10, 406 (1968).
28. MURTAGH B.A. and SARGENT R.W.H., "A Constrained Minimization Method with Quadratic Convergence", paper presented at I.M.A.-B.C.S. Joint Conference on Optimization, University of Keele, March 1968.
29. CARATHEODORY C., "Calculus of Variations and Partial Differential Equations of the First Order" Vol. 2, p. 190 (Holden-Day, 1967).
30. MURTAGH B.A. and SARGENT R.W.H., "Computational Experience with Quadratically Convergent Minimization Methods", The Computer Journal, to appear.
31. STEWART III G.W., "A Modification of Davidon's Method to Accept Difference Approximations of Derivatives "J.A.C.M. 14(1), 72 (1967).
32. GASS S.I. "Linear Programming", p. 103 (McGraw-Hill, New York 1958).
33. COLVILLE A.R. "A Comparative Study on Nonlinear Programming Codes", IBM New York Scientific Centre Tech. Rept. 320 - 2949 (June, 1968).
34. OSTROWSKI A.M., "The Solution of Equations and Systems of Equations", Pure and Applied Math. Vol. 9, pp. 195-199 (Academic Press, New York, 1966).
35. SARGENT R.W.H. and MURTAGH B.A., "The Design of Plate Distillation Columns for Multicomponent Mixtures" Trans. Inst. Chem. Engrs. 47 (4), T85, (1969).
36. WILKINSON J.H., "Rounding Errors in Algebraic Processes", Chapter 3, (H.M.S.O., London, 1963).
37. HOLLAND C.D., "Multicomponent Distillation", p. 495, 497 (Prentice Hall Inc., Englewood Cliffs N.J., 1963).

APPENDIX 1UNCONSTRAINED MINIMIZATION RESULTS

Tables 1 - 3 and 8 show the number of function evaluations required for convergence, whilst Tables 4 - 7 compare the progress of Algorithms B.3 and B.3a using Reset 2 on the first four examples.

Table 1 Method A and Fletcher Powell

Example	Method A		Fletcher-Powell	
	Algorithm 1	Algorithm 2	Algorithm 1	VA01A
1	100	62	80	34
2	123	51	64	53
3	139	125	81	53
4	206	981	257	880

Table 2 Method B

Example	Reset 1					Reset 2				
	1	2	2a	3	3a	1	2	2a	3	3a
1	93	113	94	38	64	96	93	73	60	60
2	52	46	36	45	32	51	46	36	38	40
3	82	75	64	54	35	75	73	62	41	33
4	145	193	175	70	230	153	129	113	68	130

Table 3 Example 5 Method B

n	Fletcher Powell ⁶	Reset 1					Reset 2				
		1	2	2a	3	3a	1	2	2a	3	3a
5	19	23	42	27	33	15	27	46	28	34	15
	23	25	51	31	37	19	33	52	30	39	17
10	29	45	68	52	56	49	49	83	43	58	30
	36	51	87	70	63	67	62	89	49	78	44
20	68	78	136	75	104	79	84	141	81	103	61
	84	80	140	81	112	90	95	142	85	104	65
	89	82	142	92	150	93	96	164	89	156	70
	121	95	177	132	151	141	97	188	97	168	83

TABLE 4 Rosenbrock's Valley

Step	Algorithm 3		Algorithm 3a	
	Function Evaluations	$f(\underline{x})$	Function Evaluations	$f(\underline{x})$
0	1	$2.4 \cdot 10^1$	1	$2.4 \cdot 10^1$
2	11	$5.4 \cdot 10^0$	4	$4.4 \cdot 10^0$
4	14	$2.2 \cdot 10^0$	6	$3.5 \cdot 10^0$
6	16*	$2.1 \cdot 10^0$	9	$3.1 \cdot 10^0$
8	20	$1.8 \cdot 10^0$	11	$2.5 \cdot 10^0$
10	22*	$1.4 \cdot 10^0$	14	$2.1 \cdot 10^0$
12	27	$1.1 \cdot 10^0$	16	$1.8 \cdot 10^0$
14	29	$7.2 \cdot 10^{-1}$	18	$1.5 \cdot 10^0$
16	33	$4.6 \cdot 10^{-1}$	21*	$1.2 \cdot 10^0$
18	37*	$3.8 \cdot 10^{-1}$	25	$1.0 \cdot 10^0$
20	39	$2.1 \cdot 10^{-1}$	28*	$7.6 \cdot 10^{-1}$
22	42	$1.2 \cdot 10^{-1}$	31	$5.9 \cdot 10^{-1}$
24	45*	$6.2 \cdot 10^{-2}$	34	$4.1 \cdot 10^{-1}$
26	48	$3.4 \cdot 10^{-2}$	36*	$2.8 \cdot 10^{-1}$
28	50*	$1.2 \cdot 10^{-2}$	40	$1.9 \cdot 10^{-1}$
30	52*	$4.3 \cdot 10^{-3}$	43	$9.7 \cdot 10^{-2}$
32	54	$1.5 \cdot 10^{-3}$	46	$4.2 \cdot 10^{-2}$
34	56	$8.0 \cdot 10^{-5}$	48*	$1.3 \cdot 10^{-2}$
36	58	$1.4 \cdot 10^{-7}$	51	$5.9 \cdot 10^{-3}$
38	60*	$5.3 \cdot 10^{-11}$	53*	$1.3 \cdot 10^{-3}$
40			55*	$4.3 \cdot 10^{-4}$
42			57	$6.9 \cdot 10^{-6}$
44			59	$3.3 \cdot 10^{-9}$
45			60	$2.3 \cdot 10^{-13}$

* Reset 2

TABLE 5 Quartic Singular Hessian

Step	Algorithm 3		Algorithm 3a	
	Function Evaluations	$f(\underline{x})$	Function Evaluations	$f(\underline{x})$
0	1	2.2 10^2	1	2.2 10^2
2	11	1.8 10^1	3	2.6 10^1
4	17	2.6 10^0	6	1.4 10^1
6	19	6.3 10^{-3}	8	1.1 10^0
8	21	8.5 10^{-4}	10	1.1 10^{-1}
10	23	3.9 10^{-4}	12	1.1 10^{-2}
12	25	3.0 10^{-4}	14	1.2 10^{-3}
14	28	2.4 10^{-4}	16	1.2 10^{-4}
16	30	2.3 10^{-4}	18	1.4 10^{-5}
18	32*	1.9 10^{-4}	20	2.1 10^{-6}
20	34	1.4 10^{-5}	22	9.3 10^{-7}
22	36	1.4 10^{-6}	24	8.1 10^{-7}
23	37	4.6 10^{-7}	25	7.9 10^{-7}
24	38	1.5 10^{-7}	26	7.9 10^{-7}
			28	7.7 10^{-7}
			32	7.5 10^{-7}
			34*	7.5 10^{-7}
			36	7.2 10^{-7}
			38*	7.0 10^{-7}
			40	1.0 10^{-7}

* Reset 2

TABLE 6 Helical Valley

Step	Algorithm 3		Algorithm 3a	
	Function Evaluations	$f(\underline{x})$	Function Evaluations	$f(\underline{x})$
0	1	$2.5 \cdot 10^3$	1	$2.5 \cdot 10^3$
2	6	$2.7 \cdot 10^2$	3	$1.5 \cdot 10^2$
4	9*	$9.9 \cdot 10^1$	6	$3.7 \cdot 10^1$
6	12	$1.2 \cdot 10^1$	8	$1.0 \cdot 10^1$
8	14*	$9.8 \cdot 10^0$	11	$5.1 \cdot 10^0$
10	18	$6.4 \cdot 10^0$	13*	$4.4 \cdot 10^0$
12	21*	$4.7 \cdot 10^0$	16*	$2.5 \cdot 10^0$
14	24	$3.3 \cdot 10^0$	19*	$9.5 \cdot 10^{-1}$
16	27	$1.8 \cdot 10^0$	22	$4.9 \cdot 10^{-1}$
18	29*	$5.0 \cdot 10^{-1}$	25	$1.2 \cdot 10^{-1}$
20	32	$2.1 \cdot 10^{-1}$	28	$1.2 \cdot 10^{-2}$
22	34	$4.9 \cdot 10^{-2}$	30	$9.2 \cdot 10^{-4}$
24	36	$6.4 \cdot 10^{-3}$	32	$6.2 \cdot 10^{-10}$
25	37	$4.8 \cdot 10^{-3}$	33	$3.4 \cdot 10^{-12}$
26	38	$1.9 \cdot 10^{-4}$		
28	40	$4.0 \cdot 10^{-8}$		
29	41	$3.6 \cdot 10^{-12}$		

* Reset 2

TABLE 7 Four-Dimensional Banana

Step	Algorithm 3		Algorithm 3a	
	Function Evaluations	$f(\underline{x})$	Function Evaluations	$f(\underline{x})$
0	1	1.9 10^4	1	1.9 10^4
2	17	8.6 10^1	4	1.2 10^3
4	25	2.5 10^1	6	1.6 10^2
6	27	9.1 10^0	10	4.8 10^1
8	31	4.0 10^0	13*	1.4 10^1
10	35	2.5 10^0	15	8.0 10^0
12	40*	1.3 10^0	17	7.9 10^0
14	42*	8.3 10^{-1}	19	7.9 10^0
16	45	8.2 10^{-1}	21*	7.9 10^0
18	49	5.4 10^{-1}	23*	7.9 10^0
20	51	2.9 10^{-1}	25*	7.9 10^0
22	54	1.3 10^{-1}	28	7.9 10^0
24	57	3.4 10^{-2}	31*	7.9 10^0
26	59	7.0 10^{-3}	34	7.8 10^0
28	61*	7.3 10^{-4}	37*	7.8 10^0
30	64	3.3 10^{-5}	41	7.8 10^0
32	66	2.8 10^{-7}	44	7.7 10^0
34	68	2.7 10^{-12}	46	7.6 10^0
36			49	7.5 10^0
				- - - - -

* Reset 2

Table 8 Numerical Derivatives

Example	Unmodified Stewart Algorithm	Stewart's Algorithm using Double differences
1 (B.3)	286	418
2 (B.3a)	165	201
3 (B.3a)	221	364
4 (B.3)	348	358

Example -1 Quadratic with Linear Constraints

$$\text{minimize } f^0(\underline{x}) = \sum_{j=1}^n a_j x_j + \frac{1}{2} \sum_{j=1}^n \sum_{i=1}^n c_{ij} x_i x_j$$

$$\text{subject to: } -x_i \leq 0 \quad i = 1, \dots, n$$

$$\sum_{j=1}^n g_{ij} x_j - b_i \leq 0 \quad i = n+1, \dots, m$$

$$n = 4, m = 7 \quad \text{Initial Point: } (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$$

	ij	1	2	3	4	
a_j		-1	-3	1	-1	
c_{ij}	1	2	0	-1	0	
	2	0	1	0	0	
	3	-1	0	2	1	
	4	0	0	1	1	
	ij	1	2	3	4	b_i
g_{ij}	5	1	2	1	1	5
	6	3	1	2	-1	4
	7	0	-1	-4	0	-1.5

The results are shown in Table 1 below:

Table 1.

Step	METHOD A.			METHOD B.		
	No. Function Calcns.	No. Constraints in Basis	$f^0(\underline{x})$	No. Function Calcns.	No. Constraints in Basis	$f^0(\underline{x})$
0	1	0	-1.250	1	0	-1.250
1	2	0	-2.882	2	0	-2.882
2	3	1	-4.081	3	1	-4.044
3	4	2	-4.124	4	2	-4.120
4	5	1	-4.409	5	1	-4.397
5	6	2	-4.682	6	2	-4.682

Example 2 Cubic Example of Goldfarb and Lapidus

$$\text{minimize } f^0(\underline{x}) = \sum_{j=1}^n a_j x_j + \sum_{j=1}^n \sum_{i=1}^n c_{ij} x_i x_j + \sum_{j=1}^n d_j x_j^3$$

$$\begin{aligned} \text{subject to: } & -x_i \leq 0 & i = 1, \dots, n \\ & \sum_{j=1}^n g_{ij} x_j - b_i \leq 0 & i = n+1, \dots, m \end{aligned}$$

$$n = 5, m = 15 \quad \text{Initial Point } (0,0,0,0,1)$$

	ij	1	2	3	4	5	
a_j		-15	-27	-36	-18	-12	
c_{ij}	1	30	-20	-10	32	-10	
	2	-20	39	-6	-31	32	
	3	-10	-6	10	-6	-10	
	4	32	-31	-6	39	-20	
	5	-10	32	-10	-20	30	
d_j		4	8	10	6	2	
	i^j	1	2	3	4	5	b_i
g_{ij}	6	16	-2	0	-1	0	40
	7	0	2	0	-0.4	-2	2
	8	3.5	0	-2	0	0	0.25
	9	0	2	0	4	1	4
	10	0	9	2	-1	2.8	4
	11	-2	0	4	0	0	1
	12	1	1	1	1	1	40
	13	1	2	3	2	1	60
	14	-1	-2	-3	-4	-5	-5
	15	-1	-1	-1	-1	-1	-1

The results, in comparison with Goldfarb-Lapidus, are shown in Table 2 below.

Table 2.

Step	GOLDFARB-LAPIDUS			METHOD A.			METHOD B.		
	No. Functn. Calcns.	No. Constr. in Basis	$f^0(\underline{x})$	No. Functn. Calcns.	No. Constr. in Basis	$f^0(\underline{x})$	No. Functn. Calcns.	No. Constr. in Basis	$f^0(\underline{x})$
0	1	0	+20.0000	1	0	+20.0000	1	0	+20.0000
1	2	1	-23.8967	2	1	-23.8967	2	1	-23.8967
2	3	2	-25.1972	3	2	-25.2247	3	2	-25.1909
3	5	3	-25.2605	4	2	-28.7506	4	2	-30.6838
4	6	2	-28.5235	5	3	-28.9891	5	3	-32.0117
5	7	3	-29.6326	6	3	-31.0703	6	4	-32.348664
6	8	4	-32.0615	7	3	-32.1685	7	4	-32.348679
7	10	3	-32.1134	8	3	-32.3469			
8	11	3	-32.3353	9	4	-32.348678			
9	13	4	-32.348679	10	4	-32.348679			

Example 3

This example is in 16 variables, with 8 linear equality constraints and 32 linear inequality constraints.

$$\text{Minimize } f^0(\underline{x}) = \sum_{i=1}^{16} \sum_{j=1}^{16} A_{ij} (x_i^2 + x_i + 1.0)(x_j^2 + x_j + 1.0)$$

subject to :

$$0.22x_1 + 0.20x_2 + 0.19x_3 + 0.25x_4 + 0.15x_5 + 0.11x_6 + 0.12x_7 + 0.13x_8 + x_9 = 2.5$$

$$-1.46x_1 - 1.3x_3 + 1.82x_4 - 1.15x_5 + 0.8x_7 + x_{10} = 1.1$$

$$1.29x_1 - 0.89x_2 - 1.16x_5 - 0.96x_6 - 0.49x_8 + x_{11} = -3.1$$

$$-1.10x_1 - 1.06x_2 + 0.95x_3 - 0.54x_4 - 1.78x_6 - 0.41x_7 + x_{12} = -3.5$$

$$-1.43x_4 + 1.51x_5 + 0.59x_6 - 0.33x_7 - 0.43x_8 + x_{13} = 1.3$$

$$-1.72x_2 - 0.33x_3 + 1.62x_5 + 1.24x_6 + 0.21x_7 - 0.26x_8 + x_{14} = 2.1$$

$$1.12x_1 + 0.31x_4 + 1.12x_7 - 0.36x_9 + x_{15} = 2.3$$

$$0.45x_2 + 0.26x_3 - 1.10x_4 + 0.58x_5 - 1.03x_7 + 0.1x_8 + x_{16} = -1.5$$

$$0 \leq x_i \leq 5.0, \quad i = 1 \text{ --- } 16. \text{ Initial Point(Non-Feasible): } x_i = 0$$

$$i = 1 \text{ --- } 16.$$

The coefficients of the upper triangular matrix A_{ij} are shown overleaf.

The results are shown in Table 3 below :

Table 3

Step	Method A			Method B		
	No.Functn. Calcns.	No.Constr. in Basis	$f^0(\underline{x})$	No.Functn. Calcns.	No.Constr. in Basis	$f^0(\underline{x})$
0	1	11	246.271	1	11	246.271
1	2	11	245.620	2	11	245.620
2	3	12	245.028	3	12	245.094
3	5	13	244.955	5	13	244.918
4	7	13	244.955	7	13	244.913
5	9	13	244.951	9	13	244.900
6	11	13	244.951			
7	13	13	244.950			
8	15	13	244.932			
9	16	13	244.908			
10	17	13	244.905			
11	19	13	244.901			

Example 4 Four-Dimensional Banana

This example has linear inequality constraints in the form of bounds on the variables, which are usually inactive.

$$\begin{aligned} \text{Minimize } f^0(\underline{x}) = & 100(x_1^2 - x_2)^2 + (1 - x_1)^2 + 90(x_3^2 - x_4)^2 \\ & + (1 - x_3)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] \\ & + 19.8(x_2 - 1)(x_4 - 1) \end{aligned}$$

subject to $-10.0 \leq x_i \leq 10.0 \quad i = 1 \text{ --- } 4.$

Initial point $(-3, -1, -3, -1).$

The results are shown in Table 4 below :

Table 4

Method A			Method B		
Step	No.Functn. Calcns.	$f^0(\underline{x})$	Step	No.Functn. Calcns.	$f^0(\underline{x})$
0	1	$1.9 \cdot 10^4$	0	1	$1.9 \cdot 10^4$
4	20	$2.8 \cdot 10^1$	2	6	$9.3 \cdot 10^3$
8	34	$1.5 \cdot 10^1$	4	8*	$1.8 \cdot 10^3$
12	47	$2.4 \cdot 10^0$	6	10	$6.2 \cdot 10^2$
16	58	$7.5 \cdot 10^{-1}$	8	12	$1.2 \cdot 10^2$
20	66	$7.5 \cdot 10^{-1}$	10	14	$7.6 \cdot 10^0$
24	80	$4.9 \cdot 10^{-1}$	12	16	$9.8 \cdot 10^{-1}$
28	94	$2.3 \cdot 10^{-1}$	14	19 ⁺	$8.0 \cdot 10^{-1}$
32	106	$1.1 \cdot 10^{-1}$	16	21	$8.0 \cdot 10^{-1}$
36	118	$1.1 \cdot 10^{-1}$	18	24	$5.2 \cdot 10^{-1}$
40	132	$8.6 \cdot 10^{-2}$	20	27	$3.4 \cdot 10^{-1}$
44	146	$8.4 \cdot 10^{-2}$	22	30	$1.9 \cdot 10^{-1}$
48	159	$5.1 \cdot 10^{-3}$	24	32	$8.0 \cdot 10^{-2}$
52	174	$5.0 \cdot 10^{-3}$	26	34	$2.5 \cdot 10^{-2}$
56	187	$2.8 \cdot 10^{-3}$	28	36	$5.6 \cdot 10^{-3}$
60	195	$1.5 \cdot 10^{-5}$	30	38	$2.8 \cdot 10^{-4}$
64	203	$1.0 \cdot 10^{-5}$	32	40	$1.8 \cdot 10^{-6}$
66	206	$2.6 \cdot 10^{-10}$	34	42	$2.1 \cdot 10^{-9}$
			35	43	$1.2 \cdot 10^{-12}$

* - Constraint 6 was in the basis ($x_2 = 10.0$) for steps 4 and 5.

+ - Reset 2.

Appendix 3 : Nonlinear Constraints

Example 1

$$\begin{aligned} \min. \quad f^0(\underline{x}) &= (x_1 - x_2)^2 + ((x_1 + x_2 - 10)/3)^2 + (x_3 - 5)^2 \\ \text{subject to} \quad f^1(\underline{x}) &= x_1^2 + x_2^2 + x_3^2 - 48 = 0 \\ &- 4.5 \leq x_1, x_2 \leq 4.5 \\ &- 5.0 \leq x_3 \leq 5.0 \end{aligned}$$

Initial Point (4, 4, 4)

The progress of solution is shown in Table 1 below :

TABLE 1

Step	No. Function Calcns.	No Constraint Calcns.	No. Constraints in Basis	$f^0(\underline{x})$
0	1	1	1	1.4444
1	4	3	1	1.0022
2	6	5	1	0.9545
3	8	7	1	0.9535
4	9	8	1	0.9535

Example 2

$$\begin{aligned} \min \quad f^0(\underline{x}) &= 5.3578547 x_3^2 + 0.8356891 x_1 x_5 + 37.293239 x_1 - 40792.141 \\ \text{subject to} \quad &0 \leq 85.334407 + 0.0056858 x_2 x_5 + 0.0006262 x_1 x_4 - 0.0022053 x_3 x_5 \leq 92 \\ &90 \leq 80.51249 + 0.0071317 x_2 x_5 + 0.0029955 x_1 x_2 + 0.0021813 x_3^2 \leq 110 \\ &20 \leq 9.300961 + 0.0047026 x_3 x_5 + 0.0012457 x_1 x_3 + 0.0019085 x_3 x_4 \leq 25 \end{aligned}$$

Example 2 (continued)

$$78 \leq x_1 \leq 102$$

$$33 \leq x_2 \leq 45$$

$$27 \leq x_3 \leq 45$$

$$27 \leq x_4 \leq 45$$

$$27 \leq x_5 \leq 45$$

Initial Point (78.62, 33.44, 31.07, 44.18, 35.32)

3 slack variables were added to the problem, their initial values being that of the corresponding constraint functions.

The progress of solution is shown in Table 2 below :

TABLE 2

Step	No. Function Calcns.	No. Constraint Calcns.	No. Constraints in Basis	$f^0(\underline{x})$
0	1	1	3	- 30373.9
1	2	2	3	- 30497.7
2	3	3	4	- 30556.1
3	5	5	5	- 30586.6
4	7	7	6	- 30640.8
5	9	9	8	- 30665.5

Example 3

$$\text{minimize } \sum_{i=1}^{10} b_i x_i + \sum_{i=1}^5 \sum_{j=1}^5 c_{ij} y_i y_j + 2 \sum_{j=1}^5 d_j y_j^3$$

subject to

$$a_j + 2 \sum_{i=1}^5 c_{ij} y_i + 3d_j y_j^2 + \sum_{i=1}^{10} g_{ij} x_i \geq 0 \quad j = 1 \dots 5$$

$$y_j \geq 0 \quad j = 1 \dots 5, \quad x_i \geq 0 \quad i = 1 \dots 10$$

where the elements a_j , b_i , c_{ij} , d_j and g_{ij} are as given for Example 2, Appendix 2.

Initial Point : $x_7 = 60.0$, all other variables 0.0001.

5 slack variables were added to the problem, their initial values being that of the corresponding constraint functions.

The progress of solution is shown in Table 3 below :

TABLE 3

Step	No. Function Calcns.	No. Constraint Calcns.	No. Constraints in Basis	$f^0(\underline{x})$
0	1	1	14	$24.0 \cdot 10^2$
4	21	8	19	$14.4 \cdot 10^2$
8	28	14	19	$8.4 \cdot 10^2$
12	40	23	19	71.9076
16	50	29	19	59.8411
20	70	38	19	42.2758
24	89	45	18	42.1913
28	102	58	19	33.5005
32	114	68	19	33.0912
36	126	75	18	33.0798
40	137	85	18	32.9879
44	146	99	17	32.4885
48	158	108	17	32.3570
52	168	114	16	32.3493
56	178	119	16	32.3487

Note that in all these examples, a penalty term was added, as discussed in section 4.6.3, with a value of K equal to 2.0.