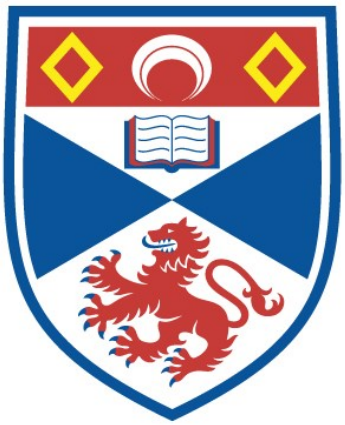
# MODIFICATIONS OF SOME ALGORITHMS FOR UNCONSTRAINED OPTIMIZATION

K. Mirnia-Harikandi

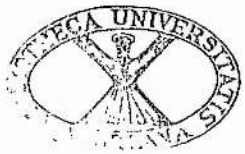
A Thesis Submitted for the Degree of PhD
at the
University of St Andrews



1979

# MODIFICATIONS OF SOME ALGORITHMS FOR UNCONSTRAINED OPTIMIZATION

K. MIRNIA-HARIKANDI




Thesis submitted for the
Degree of Doctor of Philosophy
of the University of St. Andrews



ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

CERTIFICATE

I certify that Kamal Mirnia has satisfied the conditions of the Ordinance and Regulations and is thus qualified to submit the accompanying application for the degree of Doctor of Philosophy.

## DECLARATION

I declare that the following thesis is a record of research work carried out by me, that the thesis is my own composition, and that it has not been presented in application for a higher degree previously.

K. Mirnia Harikandi

ACKNOWLEDGMENT

I wish to express my thanks to Professor S. N. Curle, Head of the Applied Mathematics Department of the University of St. Andrews for allowing me to work in the department. My sincere thanks are due to my supervisor, Mr M. A. Wolfe, for his help and encouragement.

I am indebted to my own University, the University of Azarabadegan in Iran, for allowing me leave to work in Great Britain; I am indebted also to the Ministry of Science and Higher Education which has given me great assistance.

I would also like to thank, most sincerely, my wife Victoria, for her continual encouragement and patience during the time which it took to complete the work for this thesis.

Last but not least I would like to thank to Dr J. T. Henderson for his advice and help with computing, Dr R. Fletcher who provided the listing of the program related to his joint paper with Freeman. Also, I would like to thank Mrs G. Hall who undertook the typing of this thesis and Mr H. Feiz who kindly filled in the untyped symbols.

## ABSTRACT

This thesis contains an account of several modifications to two algorithms for unconstrained optimization, both of which are due to Gill and Murray.

Chapter One contains a brief survey of unconstrained optimization and contains also some results which are used subsequently.

Chapter Two contains an account of some work on iterative procedures for the solution of operator equations in Banach spaces due to Wolfe (1978a) in which it is suggested that it may be possible, in certain circumstances, to use high-order iterative procedures rather than Newton's method, thereby obtaining computational advantages.

In Chapter Three the Newton-type algorithm of Gill and Murray (1974) is described and the ideas contained in Chapter Two are used to construct some modifications of this algorithm.

Chapter Four contains some algorithms for the numerical estimation of both full and band-type Hessian matrices. These algorithms may be used in conjunction with the optimization algorithms which are described in Chapters Three and Five.

In Chapter Five the least-squares algorithm of Gill and Murray (1976) is described and the ideas contained in Chapter Two are used to construct some modifications of this algorithm.

Chapter Six contains the computational results which were obtained by using the algorithms which are described in Chapters Three, Four and Five to solve the test problems which are listed in Appendices One and Two.

Contents

# Chapter 1.

# Unconstrained Optimization

## 1.1 Historical Survey

The existence of optimization problems is as old as mathematics. Indeed, even Euclid knew how to find the point on the straight line

$$ax + by = c \qquad (1.1.1)$$

which is closest to the origin, using the compass and straightedge; in fact he was minimizing $(x^2 + y^2)^{1/2}$ subject to the linear constraint (1.1.1). In the Seventeenth Century, Newton and Leibnitz derived the fundamental theorems of differential calculus, with which it became possible to find maximizers and minimizers of continuous functions. By the Nineteenth Century the more specialized method of Lagrange multipliers had been developed for solving optimization problems subject to equality constraints.

In 1847 Cauchy introduced the method of steepest descent for unconstrained minimization and it was subsequently used by Courant (1943), Curry (1944) and some others. The method of steepest descent was one of the few optimization methods which were supported by a convergence theory (see Goldstein (1962)). Another method which was supported by convergence theory was Newton's method which converges quadratically (see Cauchy (1829) and Traub (1964)). The slow convergence of the method of steepest descent on one hand and the need to determine the first and second order partial derivatives of the objective function in Newton's method gave rise to the development of many other unconstrained optimization methods some of which are

reviewed by Broyden (1965), Box (1966), Kowalik and Osborne (1968), Bard (1970), Huang (1970), Powell (1971), Dixon (1973), Gill and Murray (1974), Dennis and Moré (1977), Brodlie (1977) and Wolfe (1978). But in both Newton's method and in the quasi-Newton methods, some practical difficulties still remained; the most important one being the indefiniteness of the Hessian or its approximation. In the case of non-linear least-squares problems, Levenberg (1944) added a suitable multiple of the identity matrix to the approximated Hessian, as did Marquardt (1963). Greenstadt (1967) applied eigen-system analysis to Newton's method, and Fiacco and Mccormick (1968) applied Cholesky factorization to Newton's method. Murray (1972) suggested a modified Cholesky factorization which is numerically stable and provides a positive definite matrix as an approximation to the Hessian; this idea has been implemented by Gill and Murray (1974a). Also, Gill and Murray (1976) have described numerically stable methods involving the singular value decomposition of the Jacobian for solving unconstrained non-linear least squares problems.

From 1959, when Davidon described what might be called the first quasi-Newton method, until about 1970, many authors devoted their attention to the construction of efficient quasi-Newton methods. Since about 1970, however, several authors, among whom are Powell (1971a), Broyden, Dennis and Moré (1973), Dennis and Moré (1974), and Moré and Trangenstein (1976), have contributed to the convergence theory of various algorithms for unconstrained optimization.

Before about 1970, high-accuracy line-searches were used in most unconstrained optimization algorithms. Nowadays a great deal of attention is devoted to the replacement of line searches with step-length algorithms. Furthermore, much attention has recently been

given to the convergence theory of methods for unconstrained optimization in which steplength algorithms rather than line-searches are used. The reader is referred to the papers by Stoer (1975) and Powell (1975), for example.

## 1.2 Descent Methods.

An important class of methods for the unconstrained minimization of differentiable functions $F : R^n \longrightarrow R^1$ is the class of descent methods. Each member of this class is iterative and is such that the value of the objective function $F$ decreases at each iteration. In order to describe the class of descent methods in greater detail we need the following definition.

Definition 1.2.1

Suppose that $F : R^n \longrightarrow R^1$ is G - differentiable.

Then the vector $p \in R^n$ is said to be a down-hill direction for $F$ at $x \in R^n$ if and only if

$$p^T g(x) < 0 \qquad (1.2.1)$$

where $g(x) \in R^n$ is the gradient vector of $F$ at $x$. □

Theorem 1.2.1

If 1. $F : D \subset R^n \longrightarrow R^1$ is G - differentiable at $X \in \text{int}(D)$;

2. $p$ is a down-hill direction at $x$,

then $\exists\, \delta > 0$ such that ( $\forall\, \alpha \in (0, \delta)$ )

$$F(x + \alpha p) < F(x) \qquad (1.2.2)$$

Proof.

See Ortega and Rheinboldt(1970). □

Any iterative method which provides a down-hill direction $p^{(k)}$, an estimate $x^{(k)}$ of the minimizer of F, and a scalar $\alpha^{(k)}$ at stage k to satisfy (1.2.1) and (1.2.2) is referred to as a descent algorithm. The general descent algorithm is as follows.

Algorithm 1.2.1

Suppose that $F : R^n \longrightarrow R^1$ has an unconstrained local minimizer $x^* \in R^n$ and $x^{(0)} \in R^n$ is an initial estimate of $x^*$ .

step 1. Set $k = 0$ .

step 2 Find $p^{(k)}$ such that $p^{(k)T} g(x^{(k)}) < 0$.

step 3. Determine $\alpha^{(k)} \in R^1$ such that

$$F(x^{(k)} + \alpha^{(k)} p^{(k)}) < F(x^{(k)}) .$$

step 4. Set

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)} ,$$

$k = k + 1$, and go to Step 2. □

It is evident that, when $g(x^{(K)}) \neq 0$, then the set

$$S = \left\{ p \in R^n : p^T g(x^{(K)}) < 0 \right\}$$

is non-empty, since $-g(x^{(K)}) \in S$. Thus, either at stage k, $g(x^{(K)}) = 0$ which indicates that $x^{(K)}$ is a critical point of F, or otherwise, there exists a direction along which the value of the objective function is decreased by a suitable choice of $\alpha^{(K)}$. Now suppose that the sequence $(x^{(K)})$ is generated from a descent algorithm such that $(\forall k \geq 0)\ g(x^{(K)}) \neq 0$. Then the following theorem is true.

Theorem 1.2.2

If 1. $F : R^n \longrightarrow R^1$ is a given function and $F \in C^2(D)$,

where D is an open convex set;

2. $x^{(0)} \in D$;

3. $S \subset D$, where $S = \left\{ x \in R^n : F(x) \leq F(x^{(0)}) \right\}$;

4. $\exists L \in R^1$ such that $L \leq F(x)$ $(\forall x \in S)$;

5. $\exists M > 0$ such that $M \geq \| G(x) \|$ $(\forall x \in S)$;

6. $p^{(K)}$ is chosen so that

$$p^{(K)T} g^{(K)} \leq -\varepsilon_1 \| p^{(K)} \| \cdot \| g^{(K)} \| \ (\forall k \geq 0),$$

where $\varepsilon_1 > 0$ is given;

7. the sequence $(x^{(K)})$ is generated from Algorithm 1.2.1;

8. $\exists\, \varepsilon_2 \in [0,1)$ such that

$$|g^{(K+1)T} p^{(K)}| \leqslant \varepsilon_2 |g^{(K)T} p^{(K)}| \quad (\forall k \geqslant 0) ;$$

9. $\exists\, \varepsilon_3 > 0$ such that

$$F^{(K)} - F^{(K+1)} \geqslant -\varepsilon_3 \alpha^{(K)} g^{(K)T} p^{(K)} \quad (k \geqslant 0) ,$$

then corresponding to each $\varepsilon_4 > 0$, there exists a positive integer $K$ such that

$$\| g^{(K)} \| \leqslant \varepsilon_4 \qquad (k \geqslant K) .$$

That is, Algorithm 1.2.1 will terminate.

Proof.

See Wolfe (1978). □

In order to state a more general theorem related to the convergence of the sequence obtained from Algorithm 1.2.1, we follow Ortega and Rheinboldt(1970).

Definition 1.2.2

A mapping $\sigma : [0, \infty) \longrightarrow [0, \infty)$ is a <u>forcing function</u>

(or F - function) if and only if for any sequence $(t_K)$ $(t_K \in [0,\infty)(\forall k \geq 0$

$$(\lim_{K\to\infty} \sigma(t_K) = 0) \Longrightarrow (\lim_{K\to\infty} t_K = 0) \ . \ \square \qquad (1.2.3)$$

The following theorem is used subsequently.

Theorem 1.2.3

If 1. $F : R^n \longrightarrow R^1$ is G - differentiable and bounded below;

2. the sequences $(x^{(K)})$, $(p^{(K)})$, and $(\alpha^{(K)})$ are determined from Algorithm 1.2.1;

3. there exists a forcing function $\sigma : [0,\infty) \to [0,\infty)$

such that $(\forall k \geq 0)$,

$$F(x^{(K)}) - F(x^{(K)} + \alpha^{(K)} p^{(K)}) \geq \sigma(g^{(K)T} p^{(K)} / \| p^{(K)} \|) , \qquad (1.2.4)$$

then

$$g^{(K)T} p^{(K)} / \| p^{(K)} \| \longrightarrow 0 \quad (k \longrightarrow \infty).$$

Proof.

Since F is bounded below, then the sequence $(F(x^{(K)}))$ is bounded below, and by (1.2.4) it is also non-increasing. Therefore

$(F(x^{(K)}))$ is a convergent sequence and thus

$$(F(x^{(K)}) - F(x^{(K+1)})) \longrightarrow 0 \qquad (k \longrightarrow \infty) ,$$

whence from (1.2.4) and Definition 1.2.2,

$$g(x^{(K)})^T p^{(K)} / \| p^{(K)} \| \longrightarrow 0 \quad \text{as} \quad k \longrightarrow \infty . \qquad (1.2.5)$$

□

Corollary 1.2.1

If 1. the hypotheses of Theorem 1.2.3 are valid;

2. $\exists C > 0$ such that

$$-g(x^{(K)})^T p^{(K)} \geq C \| g^{(K)} \| . \| p^{(K)} \| \quad (\forall k \geq 0) , \qquad (1.2.6)$$

then

$$\lim_{K \to \infty} g(x^{(K)}) = 0 . \qquad (1.2.7)$$

Proof.

By Theorem 1.2.3, (1.2.5) holds. Therefore from (1.2.6), (1.2.7) holds. □

Definition 1.2.3

The step-length $\alpha^{(K)}$ obtained in Algorithm 1.2.1 satisfies

the condition of sufficient decrease if and only if inequality (1.2.4) is satisfied. □

Definition 1.2.4

A direction of search $p$ at $x$ is said to be a steepest descent direction for $F$ with respect to a vector norm $\| \cdot \| : R^n \longrightarrow R^1$ if and only if

$$\| g(x) \| = -g(x)^T p / \| p \| \quad . \quad \square$$

Theorem 1.2.4

If 1. $F : R^n \longrightarrow R^1$ is G - differentiable at $x$;

2. $B$ is an $n \times n$ symmetric positive definite matrix;

3. $\| \cdot \| : R^n \longrightarrow R^1$ is defined by

$$\| x \| = (x^T Bx)^{1/2} \quad (\forall x \in R^n) ,$$

then the steepest descent direction for $F$ at $x$ with respect to the norm defined by Hypothesis 3 is given by

$$p = - B^{-1} g(x) .$$

Proof.

See Ortega and Rheinboldt (1970). □

Thus, if in the descent algorithm, at stage $K$, we determine a symmetric positive definite matrix $B^{(K)}$ and set

$$p^{(K)} = - B^{(K)-1} g(x^{(K)}), \qquad (1.2.8)$$

then $p^{(K)}$ is the steepest descent direction at $x^{(K)}$ with respect to the norm $\| \cdot \|^{(K)}$ defined by $\| x \|^{(K)} = (x^T B^{(K)} x)^{1/2}$. The corresponding descent method is referred to as a variable metric method.

In the descent algorithm, having found the search direction, the step-length $\alpha^{(K)}$ has to be determined in such a way that $(x^{(K)})$ generated from the algorithm has the critical points of F as limit point under some reasonable conditions. The procedure for determining such $\alpha^{(K)}$ is called a step-length algorithm. In particular, a step-length algorithm which determines $\alpha^{(K)}$ so as to minimize $\varphi(\alpha) = F(x^{(K)} + \alpha p^{(K)})$ is referred to as a line-search. Attempts have been made by many research workers such as Curry (1944), Keifer (1953), Johnson (1955), Glayzal (1959) and Crockett and Chernoff (1955), to construct efficient line search algorithms.

Davidon (1959) has suggested that the one-variable function $\varphi(\alpha) = F(x^{(K)} + \alpha p^{(K)})$ be approximated with a cubic polynomial and $\alpha^{(K)}$ determined to be the minimizer of this polynomial. Himmelblau (1972), Gill and Murray (1974b), and Wolfe (1978) have given algorithms for line-searches which involve quadratic interpolation. Altman (1966), Goldstein (1967), and Armijo (1966) have proposed step-length algorithms details of which are given by Ortega and

Rheinboldt (1970).

Gill and Murray (1974b) have described two algorithms, namely the safeguarded cubic and quadratic univariate minimization algorithms, in which the function $\varphi$ is approximated with polynomials of degrees 3 and 2 respectively. Gill and Murray have also described two step-length algorithms, namely the Cubic and quadratic interpolation step-length algorithms, in which the cubic and quadratic univariate minimization algorithms are used.

Let $S_C = \{\alpha_I^{(K)}\}$ be the set of points generated by the safeguarded cubic univariate minimization algorithm, and let $S_Q = \{\alpha_I^{(k)}\}$ be the set of points generated by the safeguarded quadratic univariate minimization algorithm. The cubic interpolation step-length algorithm of Gill and Murray is as follows.

Algorith 1.2.2

Step 1. Compute the first member $\bar{\alpha}$ of $S_c$ such that

$$| g(x^{(K)} + \bar{\alpha} p^{(K)})^T p^{(K)} | \leq -\eta g(x^{(K)})^T p^{(K)}, \qquad (1.2.9)$$

where $\eta \in [0, 1)$ is a pre-assigned scalar

and

$$F(x^{(K)} + \bar{\alpha} p^{(K)}) < F(x^{(K)}).$$

Note: when $\eta = 0$, $\bar{\alpha}$ is a local minimizer of $F$ along the line $x^{(K)} + \alpha p^{(K)}$.

Step 2. If $\bar{\alpha}$ is such that

$$F(x^{(k)}) - F(x^{(k)} + \bar{\alpha} p^{(k)}) \geq -\mu \bar{\alpha} g(x^{(k)})^T p^{(k)},$$

where $\mu \in (0, \frac{1}{2}]$ is another pre-assigned scalar, then set $\alpha^{(k)} = \bar{\alpha}$. Otherwise, let $w$ be the first member of the set $\{2^{-j} : j \geq 1\}$ such that $w\bar{\alpha}$ satisfies

$$F(x^{(k)}) - F(x^{(k)} + w\bar{\alpha} p^{(k)}) \geq -\mu w \bar{\alpha} g(x^{(k)})^T p^{(k)}.$$

Set $\alpha^{(k)} = w\bar{\alpha}$. □

The quadratic interpolation step-length algorithm of Gill and Murray is as follows.

Algorithm 1.2.3

Step 1. Compute the first member $\bar{\alpha}$ of $S_Q$ such that

$$\left| \frac{F(x^{(k)} + \alpha_l^{(k)} p^{(k)}) - F(x^{(k)} + \bar{\alpha} p^{(k)})}{\alpha_l - \bar{\alpha}} \right| \leq -\eta\, g(x^{(k)})^T p^{(k)}, \tag{1.2.10}$$

and

$$F(x^{(k)} + \bar{\alpha} p^{(k)}) < F(x^{(k)}),$$

where $\alpha_l$ is the last member of $S_Q$ for which $\alpha_l < \bar{\alpha}$, and $\eta \in [0, 1)$ is a pre-assigned scalar.

Step 2. If $\bar{\alpha}$ is such that

$$F(x^{(k)}) - F(x^{(k)} + \bar{\alpha} p^{(k)}) \geq -\mu \bar{\alpha} g(x^{(k)})^T p^{(k)},$$

where $\mu \in (0, \frac{1}{2}]$ is another pre-assigned scalar, then set $\alpha^{(K)} = \bar{\alpha}$.

Otherwise let $w$ be the first member of the set $\left\{ 2^{-j} : j \geqslant 1 \right\}$ such that $w\bar{\alpha}$ satisfies

$$F(x^{(K)}) - F(x^{(K)} + w\bar{\alpha} p^{(K)}) \geqslant -\mu w \bar{\alpha} g(x^{(K)})^T p^{(K)} .$$

Set $\alpha^{(K)} = w\bar{\alpha}$. □

The following theorems show that algorithms 1.2.2 and 1.2.3 satisfy the condition of sufficient decrease.

Theorem 1.2.5

If 1. $F : D \subset R^n \longrightarrow R^1$ and $F \in C^1(D)$, where D is an open set;

2. $\bar{\Omega}(F(x^{(0)})) = \left\{ x : x \in D \text{ and } F(x) \leqslant F(x^{(0)}) \right\}$ is compact;

3. $CO\left[\bar{\Omega}(F(x^{(0)}))\right] \subset D$ where $CO(\bar{\Omega})$ denotes the closed convex hull of $\bar{\Omega}$;

4. $(p^{(K)})$ is any sequence of down-hill direction for F,

then the sequence $(\alpha^{(K)})$ generated from Algorithm 1.2.2 satisfies the condition of sufficient decrease.

Proof.

See Gill and Murray (1974b). □

Theorem 1.2.6

If 1. the hypotheses of Theorem 1.2.5 are valid;

2. $( \alpha^{(K)} )$ is the sequence generated from Algorithm 1.2.3,

then the sequence $( \alpha^{(K)} )$ satisfies the condition of sufficient decrease.

Proof.

See Gill and Murray (1974b). □

In practice to prevent possible overflow, we may have to restrict the $\alpha^{(K)}$ generated from the univariate search according to $\alpha^{(K)} \leqslant \lambda$ for a fixed $\lambda$. Thus, the value of $\bar{\alpha}$ obtained may be such that $\bar{\alpha} = \lambda$ before the conditions (1.2.9) or (1.2.10) met. In this event, the following theorem is valid.

Theorem 1.2.7

If 1. the hypotheses of Theorem 1.2.5 are valid;

2. the sets $S_Q$ and $S_C$ are such that

$$\alpha^{(K)} \leqslant \lambda \qquad (\forall k \geqslant 0),$$

then the step-length $\alpha^{(K)}$ generated from both algorithms 1.2.2 and 1.2.3 satisfies either the sufficient decrease conditions or is such that

$$F(x^{(K)}) - F(x^{(K)} + \alpha^{(K)} p^{(K)}) > -\mu\lambda \; g(x^{(K)})^T p^{(K)} .$$

Proof.

See Gill and Murray (1974b). □

In subsequent sections we describe some algorithms which are members of the class of descent methods which are obtained by choosing a special vector as a down-hill direction.

An important concept in connection with iterative methods for unconstrained optimization is the rate of convergence of the sequence $( x^{(K)} )$ of estimates of a minimizer.

Definition 1.2.5

Suppose that, in a normed linear space, the sequence $(x^{(K)})$ converges to $x^*$ and $x^{(K)} \neq x^* \; (\forall k \geqslant K)$ for some K. Then the order of convergence of $(x^{(K)})$ with respect to the norm $\| . \|$ is the largest integer p $(p \geqslant 1)$ such that, for some $C \in (0, \infty)$,

$$\lim_{K \to \infty} \frac{\|x^{(K+1)} - x^*\|}{\|x^{(K)} - x^*\|^p} = C . \qquad (1.2.11)$$

□

Definition 1.2.6

Suppose that, in a normed linear space, the sequence $(x^{(K)})$ converges to $x^*$, and that $x^{(K)} \neq x^* \; (\forall k \geqslant K)$ for some K. Then $(x^{(K)})$ converges Q - linearly if and only if for some $C \in (0, \infty)$

$$\lim_{K \to \infty} \frac{\|x^{(K+1)} - x^*\|}{\|x^{(K)} - x^*\|} = C , \qquad (1.2.12)$$

and $(x^{(k)})$ converges $\underline{Q}$ - $\underline{\text{superlinearly}}$ if and only if

$$\lim_{k\to\infty} \frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} = 0 . \qquad (1.2.13)$$ □

Usually, the more efficient methods are Q-superlinearly convergent.

For more detail about convergence rates of iterative procedures see Ortega and Rheinboldt(1970) Chapter 9.

The following results are quoted from Dennis and Moré (1974) when the direction of search is obtained as in (1.2.8).

Lemma 1.2.1

If 1. $(x^{(k)})$ is a sequence in $R^n$ ;

2. $(x^{(k)})$ converges to $x^*$ Q - superlinearly,

then

$$\lim_{k\to\infty} \frac{\|x^{(k+1)} - x^{(k)}\|}{\|x^{(k)} - x^*\|} = 1 .$$

Proof.

See Dennis and Moré (1974). □

Theorem 1.2.8

Suppose that

1. $F : R^n \longrightarrow R^1$ and $F \in C^1(D)$ where $D \subseteq R^n$

is an open convex set;

2. $\exists x^* \in D$ such that the Hessian G of F is continuous at $x^*$ and $G(x^*)$ is non singular;

3. $(B^{(k)})$ is a sequence of non-singular matrices;

4. $x^{(0)} \in D$ ;

5. the sequence $(x^{(k)})$ generated from

$$x^{(k+1)} = x^{(k)} - B^{(k)-1} g^{(k)} \qquad (k \geq 0) \qquad (1.2.14)$$

remains in D and converges to $x^*$ .

Then $(x^{(k)})$ converges to $x^*$ Q - superlinearly and $g(x^*) = 0$ if and only if

$$\lim_{k \to \infty} \frac{\| [B^{(k)} - G(x^*)] (x^{(k+1)} - x^{(k)}) \|}{\| x^{(k+1)} - x^{(k)} \|} = 0 . \qquad (1.2.15)$$

Proof.

See Dennis and Moré (1974). □

Theorem 1.2.9

Suppose that

1. hypotheses 1 - 4 of Theorem 1.2.8 are valid;

2. the sequence $(x^{(k)})$ generated from

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} B^{(k)-1} g^{(k)}$$

remains in D and converges to $x^*$ ;

3. (1.2.15) holds.

Then $(x^{(k)})$ converges to $x^*$ Q - superlinearly and $g(x^*) = 0$

if and only if

$$\lim_{k \to \infty} \alpha^{(k)} = 1 .$$

Proof.

See Dennis and Moré (1974). □

The following results do not appear in any source which is known to the Author.

Lemma 1.2.2

Suppose that

1. the sequence $(\delta^{(k)})$ in $R^n$ is bounded and

$$\delta^{(k)} \neq 0 \quad (\forall k \geq 0) ;$$

2. the sequence of non-singular $n \times n$ matrices $(B^{(k)})$ satisfies

$$\lim_{k \to \infty} (B^{(k+1)} - B^{(k)}) = 0 ; \tag{1.2.16}$$

3. A is any $n \times n$ matrix,

Then

$$\lim_{k \to \infty} \frac{\| [B^{(k)} - A] \delta^{(k+1)} \|}{\| \delta^{(k+1)} \|} = 0 \tag{1.2.17}$$

if and only if

$$\lim_{k \to \infty} \frac{\| (B^{(k+1)} - A) \delta^{(k+1)} \|}{\| \delta^{(k+1)} \|} = 0 . \tag{1.2.18}$$

Proof.

Because

$$(B^{(K+1)} - A)\,\delta^{(K+1)} = (B^{(K)} - A)\,\delta^{(K+1)} + (B^{(K+1)} - B^{(K)})\,\delta^{(K+1)}$$

it follows that

$$\left| \frac{\|(B^{(K)} - A)\,\delta^{(K+1)}\|}{\|\delta^{(K+1)}\|} - \frac{\|(B^{(K+1)} - B^{(K)})\,\delta^{(K+1)}\|}{\|\delta^{(K+1)}\|} \right|$$

$$\leq \frac{\|(B^{(K+1)} - A)\,\delta^{(K+1)}\|}{\|\delta^{(K+1)}\|}$$

$$\leq \frac{\|(B^{(K)} - A)\,\delta^{(K+1)}\|}{\|\delta^{(K+1)}\|} + \frac{\|(B^{(K+1)} - B^{(K)})\,\delta^{(K+1)}\|}{\|\delta^{(K+1)}\|} \qquad (1.2.19)$$

But, since

$$\frac{\|(B^{(K+1)} - B^{(K)})\,\delta^{(K+1)}\|}{\|\delta^{(K+1)}\|} \leq \|B^{(K+1)} - B^{(K)}\|,$$

then by (1.2.16)

$$\lim_{k \to \infty} \frac{\|(B^{(K+1)} - B^{(K)})\,\delta^{(K+1)}\|}{\|\delta^{(K+1)}\|} = 0. \qquad (1.2.20)$$

If (1.2.17) holds, then by (1.2.20), the right hand side of (1.2.19)

tends to zero as $k \to \infty$. Therefore, (1.2.18) holds. Conversely, if (1.2.18) holds then the left hand side of (1.2.19) tends to zero as $k \to \infty$ and thus, by (1.2.20), (1.2.17) holds. □

Theorem 1.2.10

Suppose that

1. the hypotheses of Theorem 1.2.8 hold;
2. the sequence $(B^{(k)})$ satisfies (1.2.16)

Then the sequence $(x^{(k)})$ converges to $x^*$ Q - superlinearly and $g(x^*) = 0$ if and only if

$$\lim_{k \to \infty} \frac{\| (B^{(k+1)} - G(x^*))(x^{(k+1)} - x^{(k)}) \|}{\| x^{(k+1)} - x^{(k)} \|} = 0. \quad (1.2.21)$$

Proof.

Set $A = G(x^*)$ and $\delta^{(k+1)} = x^{(k+1)} - x^{(k)}$.

The result then follows from Lemma 1.2.2 and Theorem 1.2.8. □

Theorem 1.2.11

If 1. the hypotheses 1, and 2 of Theorem 1.2.9 hold;

2. (1.2.21) holds;
3. the sequence $(B^{(k)})$ satisfies (1.2.16),

then the conclusion of Theorem 1.2.9 holds.

Proof.

From hypotheses 3, and 2 and Lemma 1.2.2, (1.2.15) holds.

Therefore, if $(x^{(k)})$ converges to $x^*$ Q - superlinearly then by Theorem 1.2.9,

$$\alpha^{(k)} \longrightarrow 1 \qquad (k \longrightarrow \infty) .$$

Conversely, if $\alpha^{(k)} \longrightarrow 1$ $(k \longrightarrow \infty)$ then, since (1.2.15) holds, the result follows from Theorem 1.2.9. □

## 1.3 The Method of Newton.

A subclass of descent methods is the class of steepest descent methods as described in Theorem 1.2.4. In this subclass, the different members are distinguished by different methods of choosing the $B^{(k)}$ in Equation (1.2.8). One way of choosing $B^{(k)}$ is suggested by the following three theorems.

Theorem 1.3.1

If 1. $F : D \subset R^n \longrightarrow R^1$ is a given function and $F \in C^2(D)$, where D is an open convex set;

2. $x^* \in D$ ;

3. $g(x^*) = 0$;

4. $\exists r > 0$ such that $B(x^*, r) \subset D$ and $(\forall x \in B(x^*, r))$, $G(x)$ is positive definite,

then F has a strong local minimizer at $x^*$.

Proof.

See Avriel (1976). □ ✓

pg 13.

Theorem 1.3.2

If 1. Hypothesis 1 of Theorem 1.3.1 is valid;

2. $x^* \in D$;

3. F has a local minimizer at $x^*$,

then $g(x^*) = 0$ and $G(x^*)$ is a positive semi-definite matrix.

Proof.

See Avriel (1976). □

Theorem 1.3.3

If 1. $G : D \subset R^n \longrightarrow L(R^n)$ is continuous at $x^* \in D$;

2. $G(x^*)^{-1}$ exists,

then

(a) $\exists r > 0$, and $\exists M$ such that $(\forall x \in D \cap B[x^*, r])$

$G(x)$ is non-singular and $\|G(x)^{-1}\| \leqslant M$;

(b) $G(x)^{-1}$ is continuous in x at $x^*$.

Proof.

See Ortega and Rheinboldt (1970). □

Theorem 1.3.4

If the hypotheses of Theorem 1.3.2 are valid, then $\exists r > 0$ such that $G(x)$ is positive semi-definite $(\forall x \in B(x^*, r) \cap D)$.

Proof.

See Simmons (1975). □

By Theorem 1.3.4, if $x^* \in D$ is a local minimizer of $F : D \subset R^n \longrightarrow R^1$, where $D \subset R^n$ is an open set, the Hessian G of F is continuous at $x^*$, and $G(x^*)$ is positive-definite, then $G(x)$ is positive definite in a neighbourhood of $x^*$. Thus, when x is sufficiently close to $x^*$, then we may determine $B^{(K)}$ for use in (1.2.8) from $B^{(K)} = G(x^{(K)})$. The corresponding descent method consists of generating $(x^{(K)})$ from

$$x^{(K+1)} = x^{(K)} - \alpha^{(K)} B^{(K)-1} g^{(K)} \quad (k \geqslant 0) \qquad (1.3.1)$$

in which

$$B^{(K)} = G^{(K)} . \qquad (k \geqslant 0) \qquad (1.3.2)$$

Such a descent method is referred to as a <u>modified</u> Newton Method. If $x^{(K)} \longrightarrow x^*$ $(k \longrightarrow \infty)$, then because G is continuous at $x^*$, we have

$$\lim_{k \to \infty} B^{(K)} = \lim_{K \to \infty} G(x^{(K)})$$
$$= G(x^*) .$$

Thus, if $\alpha^{(K)} \longrightarrow 1$ $(k \longrightarrow \infty)$, then by Theorem 1.2.9, or Theorem 1.2.11 the convergence of $(x^{(K)})$ is Q - superlinear. This leads to the conjuncture that $\alpha^{(K)}$ may be set to Unity for all k $(k \geqslant 0)$ when $x^{(0)}$ is sufficiently close to $x^*$. If in (1.3.1), $\alpha^{(K)} = 1$ $(\forall k \geqslant 0)$, the corresponding descent method is called <u>Newton's Method</u>.

In both the modified Newton Method and the Newton Method, it is necessary that $x^{(0)}$ be sufficiently close to $x^*$ in the sense that

$x^{(0)} \in B(x^*, r)$ where $r > 0$ is such that $G(x)$ is positive definite $\forall x \in B(x^*, r)$. In practice however, it is very difficult to decide whether or not a given initial iterate $x^{(0)}$ satisfies this condition. Thus, if $x$ is such that $G(x)$ is not positive definite, then either $G(x)^{-1}$ may not exist or the direction $p$ generated from

$$p = -G(x)^{-1} g(x)$$

may not be a down-hill direction. Furthermore, in practice, we may have to deal with objective functions for which the Hessian either is not available analytically or is computationally expensive to evaluate. Many authors, including Goldstein and Price (1967), Greenstadt (1967), Fiacco and Mccormick (1968), Mathews and Davies (1971), Murray (1972), and Fletcher and Freeman (1977) have described methods for guaranteeing a down-hill direction at each iteration. To overcome the objection that the Hessian is not available analytically or is computationally expensive to evaluate, Davidon (1959) has introduced an idea which will be surveyed in subsequent sections. In addition, Gill and Murray (1974a), in their implementation of the idea of Murray (1972) have used a finite-difference approximation to the Hessian.

## 1.4. The Gauss-Newton Method

Let $f : R^n \longrightarrow R^m$ be a given mapping and let $F : R^n \longrightarrow R^1$ be defined by

$$F(x) = \sum_{i=1}^{m} f_i(x)^2$$

$$= f(x)^T f(x) \qquad (\forall x \in R^n ; m \geq n) .$$

Let $A: R^n \longrightarrow L(R^n, R^m)$ be defined by

$$A(x) = (\partial_j f_i(x))_{m \times n} \quad (i = 1,\ldots,m;\ j = 1,\ldots,n).$$

Then the gradient vector $g(x)$ of $F$ at $x$ is given by

$$g(x) = 2A(x)^T f(x),$$

and the Hessian matrix $G(x)$ of $F$ at $x$ is given by

$$G(x) = 2(A(x)^T A(x) + \sum_{i=1}^{m} f_i(x)\, G_i(x)), \qquad (1.4.1)$$

where $G_l(x) = (\partial_j \partial_i f_l(x))_{n \times n}$ $(l = 1, 2, \ldots, m)$.

The search direction $p$ at $x$, corresponding to Newton's method for minimizing $F$ is given by

$$P = -(A(x)^T A(x) + \sum_{i=1}^{m} f_i(x) G_i(x))^{-1} A(x)^T f(x) \qquad (1.4.2)$$

When $f : R^n \longrightarrow R^m$ is linear, or $\| \sum_{i=1}^{m} f_i(x) G_i(x) \|$ is negligible compared with $\| A(x)^T A(x) \|$, then the term in (1.4.2) containing $G_i(x)$ may be negelected to give the search direction $\bar{p}$ defined by

$$\bar{p} = -(A(x)^T A(x))^{-1} A(x)^T f(x).$$

The vector $\bar{p}$ is called the Gauss-Newton or Gauss search direction. The descent method corresponding to the search direction $\bar{p}$ is referred to as the Gauss-Newton method. When $f_i$ $(1 \leqslant i \leqslant m)$ is a linear function,

we have $\bar{p} = p$. Thus, in this case, the Gauss-Newton Method is equivalent to the Newton Method.

It may well be, however, that $2A(x)^T A(x)$ is not an adequate approximation to $G(x)$ in (1.4.1) or that $A(x)^T A(x)$ is not positive definite.

Levenberge (1944) and Marquardt (1963) have described methods for determining a direction which lies between p and $-g$ by solving

$$(A(x)^T A(x) + \lambda I)\,\bar{\bar{p}} = -A(x)^T f(x)\,, \qquad (1.4.3)$$

where the scalar $\lambda$ is adjusted at each iteration. Obviously for $\lambda > 0$, $\bar{\bar{p}}$ is uniquely determined.

In order to analyse the algorithm, we require additional notation. Let V be the $n \times (n-t)$ matrix the columns of which span the null space of $A(x)^T A(x)$, and is such that $V^T V = I_{n-t}$ where $I_{n-t}$ is the unit matrix of order $n - t$. Let W be the $n \times t$ matrix the columns of which span the range of $A(x)^T A(x)$, and is such that $W^T W = I_t$. Then $A(x)\,V = 0$ and $W^T V = 0$. Since any vector in $R^n$ can be expressed as a linear combination of the columns of V and W, we have

$$\bar{\bar{p}} = \bar{\bar{p}}_1 + \bar{\bar{p}}_2\,,$$

where $\bar{\bar{p}}_1 = W\bar{\bar{u}}$, $\bar{\bar{p}}_2 = V\bar{\bar{y}}$, $\bar{\bar{u}}$ is $t \times 1$ and $\bar{\bar{y}}$ is $(n - t) \times 1$. Substituting in (1.4.3) for $\bar{\bar{p}}$ we obtain

$$A(x)^T A(x)\,W\bar{\bar{u}} + \lambda W\bar{\bar{u}} + \lambda V\bar{\bar{y}} = -A(x)^T f\,. \qquad (1.4.4)$$

Premultiplying (1.4.4) by $W^T$ we obtain

$$(W^T A(x)^T A(x) W + \lambda I)\,\bar{\bar{u}} = - W^T A(x)^T f ,$$

which uniquely defines $\bar{\bar{u}}$ . Premultiplying (1.4.4) by $V^T$ we obtain

$$\bar{\bar{y}} = 0.$$

Hence, we have $\bar{\bar{p}} = W\bar{\bar{u}} = \bar{\bar{p}}_1$ . Therefore $\bar{\bar{p}}$ is in the range of $A(x)^T A(x)$. Now let

$$\varepsilon B = \sum_{i=1}^{m} f_i \, G_i \, (x)$$

with $\| B \| = 1$ .

Then, from (1.4.2) we have

$$(A(x)^T A(x) + \varepsilon B)\, p = - A(x)^T f(x) . \qquad (1.4.5)$$

Let

$$p = p_1 + p_2 ,$$

where $p_1 = W u$ , and $p_2 = Vy$. By substituting in (1.4.5) we have

$$V^T BVy = - V^T BW u ,$$

so that in general p is not in the range of $A(x)^T A(x)$. Since $\| y \|$ is not necessarily small compared with $\| u \|$ the vectors $\bar{\bar{p}}$ and p will not be similar. In particular, when $\varepsilon$ is a large number compared

with $\| A(x)^T A(x) \|$ , then $\bar{p}$ (or $\bar{\bar{p}}$) is not an adequate approximation to $p$. This is one source of possible failure for the Gauss-Newton Method.

## 1.5 Quasi-Newton Methods.

When analytical formulae for the second order partial derivatives of the objective function $F : R^n \longrightarrow R^1$ are not available, or are computationally expensive to evaluate, quasi-Newton methods for minimizing $F$ are effective. The general quasi-Newton method is contained in the following algorithm.

Algorithm 1.5.1

Let an estimate $x^{(0)}$ of minimizer $x^*$ of $F$ and a symmetric positive definite matrix $H^{(0)}$ which is an estimate of the inverse Hessian of $F$ at $x^{(0)}$ be given.

Step 1. Set $k = 0$.

Step 2. Compute $F^{(k)}$ and $g^{(k)}$ from

$$F^{(k)} = F(x^{(k)})$$

and

$$g^{(k)} = g(x^{(k)}),$$

where $g(x^{(k)})$ is the gradient of $F$ at $x^{(k)}$.

Step 3. Compute $p^{(k)}$ from

$$p^{(k)} = - H^{(k)} g^{(k)}.$$

Step 4. Compute $\alpha^{(k)}$ by using Algorithm 1.2.2.

Step 5. Compute $x^{(k+1)}$ from

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}.$$

Step 6. Compute $g^{(k+1)}$, $s^{(k)}$ and $y^{(k)}$ from

$$g^{(k+1)} = g(x^{(k+1)}),$$

$$s^{(k)} = x^{(k+1)} - x^{(k)},$$

and

$$y^{(k)} = g^{(k+1)} - g^{(k)}.$$

Step 7. Compute $H^{(k+1)}$ from

$$H^{(k+1)} = H^{(k)} + C^{(k)}, \tag{1.5.1}$$

where the matrix $C^{(k)}$ is such that $H^{(k+1)}$ is symmetric positive definite and satisfies the quasi-Newton equation

$$H^{(k+1)} y^{(k)} = s^{(k)}. \tag{1.5.2}$$

Step 8. Set $k = k + 1$ and go to Step 3. □

The idea of a quasi-Newton method was originally introduced by Davidon (1959), and has been clarified and modified by Fletcher and Powell (1963). Since then a large amount of research has been done in this area. In particular, Broyden (1967), (1970), Fletcher (1970), Shanno (1970), Goldfarb (1970), Huang (1970), Bard (1968), Dixon (1972), Gill and Murray (1972), Davidon (1975), Dennis and Moré (1977) and Brodlie (1977) have made major contributions to the development of

quasi-Newton methods.

Since Bard (1968) observed that successive approximations of the inverse Hessian in the implementation of Fletcher and Powell (1963) may not remain positive definite, even if in theory they should be, a number of alternative updating formulae and implementations have been suggested.

Broyden (1970), Fletcher (1970), Goldfarb (1970), and Shanno (1970) have introduced independently the BFGS formula for updating an approximation to the Hessian matrix. This formula is

$$B^{(k+1)} = B^{(k)} + y^{(k)} y^{(k)T} / y^{(k)T} s^{(k)} - B^{(k)} s^{(k)} s^{(k)T} B^{(k)} / s^{(k)T} B^{(k)} s^{(k)}$$

$$(\forall k \geqslant 0), \quad (1.5.3)$$

where $B^{(0)}$ is a given symmetric positive definite matrix as an initial approximation to the Hessian.

Gill and Murray (1972) have introduced a general updating formula of the form

$$B^{(k+1)} = B^{(k)} + \left[ g^{(k+1)} \vdots g^{(k)} \right] \begin{pmatrix} \gamma_1 & \gamma_2 \\ \gamma_2 & \gamma_3 \end{pmatrix} \begin{bmatrix} g^{(k+1)T} \\ \hdashline g^{(k)T} \end{bmatrix}$$

$$(\forall k \geqslant 0), \quad (1.5.4)$$

where $\gamma_1$, $\gamma_2$ and $\gamma_3$ are real numbers, which contains updating formula (1.5.3) as a special case. Also, in the same paper, Gill and Murray have described an implementation of quasi-Newton methods corresponding to the updating formulae contained in class (1.5.4). In the implementation of quasi-Newton methods due to Gill and Murray, the system of linear equations

$$B^{(K)} p^{(K)} = - g^{(K)}, \qquad (1.5.5)$$

where $g^{(K)}$ is the gradient of the objective function F at $x^{(K)}$, must be solved for the search direction $p^{(K)}$. Gill and Murray have devised two methods for updating the Cholesky factors of $B^{(K)}$ so that at iteration k

$$B^{(K)} = L^{(K)} D^{(K)} L^{(K)T},$$

where $L^{(K)}$ is a unit lower triangular matrix and $D^{(K)}$ is a diagonal matrix. The vector $p^{(K)}$ can be determined by solving firstly

$$L^{(K)} v = - g^{(K)}$$

and then

$$L^{(K)T} p^{(K)} = - D^{(K)-1} v$$

so that if

$$L^{(K)} = ( l_{ij}^{(K)} ) \ (i, j = 1, 2, \ldots, n),$$

and

$$D^{(K)} = \text{Diag} \ (d_1^{(K)}, \ldots, d_n^{(K)}),$$

then

$$v_1 = - g_1^{(K)} \qquad (1.5.6)$$

$$v_i = - g_i^{(K)} - \sum_{j=1}^{i-1} l_{ij}^{(K)} v_j \ (i = 2, 3, .., n) \qquad (1.5.7)$$

$$p_n^{(K)} = v_n / d_n^{(K)} \qquad (1.5.8)$$

and

$$p_i^{(K)} = v_i / d_i^{(K)} - \sum_{j=i+1}^{n} l_{ji}^{(K)} p_j^{(K)} \quad (i = n-1, \ldots, 1) \quad (1.5.9)$$

The implementation of quasi-Newton methods due to Gill and Murray is contained in the following algorithm.

Algorithm 1.5.2

It is assumed that $x^{(0)}$, $L^{(0)}$ and $D^{(0)}$ are given.

Step 1. Set $k = 0$.

Step 2. Compute $F^{(K)} = F(x^{(K)})$, and $g^{(K)} = g(x^{(K)})$.

Step 3. Compute $P^{(K)}$ by solving (1.5.5) and using (1.5.6) - (1.5.9).

Step 4. Compute $x^{(K+1)}$, $F^{(K+1)}$ and $g^{(K+1)}$ from

$$x^{(K+1)} = x^{(K)} + \alpha^{(K)} P^{(K)},$$

$$F^{(K+1)} = F(x^{(K+1)}),$$

$$g^{(K+1)} = g(x^{(K+1)}),$$

by using Algorithm 1.2.2.

Step 5. Determine $B^{(k+1)}$ in the form

$$B^{(k+1)} = L^{(k+1)} D^{(k+1)} L^{(k+1)T}$$

from (1.5.4), where $B^{(k)} = L^{(k)} D^{(k)} L^{(k)T}$.

Step 6. Set $k = k + 1$ and go to Step 3. □

The following theorem contains sufficient conditions for the sequence $(x^{(k)})$ generated from a quasi-Newton Method with the BFGS updating formula to converge to a critical point of F.

Theorem 1.5.1

If 1. $F : R^n \longrightarrow R^1$ is twice-differentiable;

2. F is convex;

3. $x^{(0)} \in R^n$;

4. $L(x^{(0)}) = \left\{ x \in R^n : F(x) \leqslant F(x^{(0)}) \right\}$

is a bounded set;

5. the sequence $(x^{(k)})$ is generated from Algorithm 1.5.2 with the BFGS updating formula, using Algorithm 1.2.2;

6. $B^{(0)} \in L(R^n)$ is a symmetric positive definite matrix;

7. $\varepsilon > 0$ is any arbitrary small number,

then $\exists K$ such that

$$\| g^{(K)} \| \leqslant \varepsilon \qquad (\forall K \geqslant K).$$

Proof.

See Powell (1976). □

An implementation of Algorithm 1.5.2 with the updating formula 1.5.3 is available in the Numerical Algorithm Group (NAG) library of programs, and will be referred to as Algorithm 1.5.3.

## 1.6 A Simple Procedure for Step-length Determination.

Powell (1975) has discussed the convergence of a broad class of unconstrained optimization algorithms. The following algorithm is contained in the class which Powell has considered.

Algorithm 1.6.1

Let $x^{(0)}$ be an estimate of a minimizer $x^*$ of F, $\bar{\Delta} > 0$, $\varepsilon > 0$, $0 < c_1 < 1$, $1 \leqslant c_2$, and $0 < c_3 < 1$ be given.

Step 1. Set $k = 0$, $\Delta^{(k)} = \bar{\Delta}$, and compute $F^{(k)} = F(x^{(k)})$;

Step 2. Compute $g^{(k)} = g(x^{(k)})$.

Step 3. If $\| g^{(k)} \| \leqslant \varepsilon$ then stop.

Step 4. Compute an $n \times n$ symmetric positive definite matrix $B^{(k)}$.

Step 5. Compute $\delta^{(K)} = -B^{(K)-1} g^{(K)}$.

Step. 6 If $\| \delta^{(K)} \| \leq \Delta^{(K)}$ then go to Step 8.

Step 7. Set $\delta^{(K)} = \Delta^{(K)} \delta^{(K)} / \| \delta^{(K)} \|$.

Step 8. Set $\bar{x}^{(K)} = x^{(K)} + \delta^{(K)}$.

Step 9. Compute $\bar{F}^{(K)} = F(\bar{x}^{(K)})$.

Step 10. If $\bar{F}^{(K)} \geq F^{(K)}$ then set $\Delta^{(K)} = C_3 \| \delta^{(K)} \|$ and

go to Step 6.

Step 11. Compute $\bar{\varphi}^{(K)} = \varphi(\bar{x}^{(K)})$ where

$$\varphi(x^{(K)} + \delta) \stackrel{D}{=} F^{(K)} + g^{(K)T} \delta + \tfrac{1}{2} \delta^T B^{(K)} \delta .$$

Step 12. If $F^{(K)} - \bar{F}^{(K)} < C_1 (F^{(K)} - \bar{\varphi}^{(K)})$ then

set $\Delta^{(K+1)} = C_3 \| \delta^{(K)} \|$ and go to Step 15.

Step 13. Set $\Delta^{(K+1)} = C_2 \| \delta^{(K)} \|$.

Step 14. If $\Delta^{(K+1)} > \bar{\Delta}$ then set $\Delta^{(K+1)} = \bar{\Delta}$.

Step 15. Set $x^{(K+1)} = \bar{x}^{(K)}$, $F^{(K+1)} = \bar{F}^{(K)}$, $k = k + 1$

and go to Step 2. □

Steps 6 - 14 contain a procedure for determining the step-length. This procedure will be referred to as Powell's step-length algorithm.

The following theorem contains sufficient conditions to ensure that the sequence $(g^{(K)})$ generated from Algorithm 1.6.1 is not bounded away from zero.

Theorem 1.6.1

If 1. $F : R^n \longrightarrow R^1$ is bounded below;

2. F is differentiable;

3. $g : R^n \longrightarrow R^n$ is uniformly continuous;

4. the sequence $(x^{(K)})$ is generated from Algorithm 1.6.1,

then $g^{(K)}$ is not bounded away from zero ( $\forall K \geqslant 0$ ).

Proof.

See Powell (1975). □

The following theorem contains sufficient conditions for the Q - superlinear convergence of a sequence generated from Algorithm 1.6.1.

Theorem 1.6.2

If 1. the sequence $(x^{(K)})$ generated from Algorithm 1.6.1 converges to a limit point $x^*$ ;

2. $\exists r > 0$ such that the $\partial_i \partial_j F$ are continuous in $B[x^*, r]$ ;

3. $G(x^*)$ is positive definite;

$$4.\quad \| g(x^{(k)} + \delta^{(k)}) - g(x^{(k)}) - B^{(k)}\delta^{(k)} \| / \| \delta^{(k)} \| \to 0 \quad (k \to \infty), \tag{1.6.1}$$

where $\delta^{(k)}$ and $B^{(k)}$ are generated from Algorithm 1.6.1, then $(x^{(k)})$ converges to $x^*$ Q - superlinearly.

Proof.

See Powell (1975). □

# Chapter 2

## Iterative Methods for Solving Systems of Nonlinear Equations.

In this chapter we discuss some iterative methods on which the unconstrained minimization algorithms appearing in subsequent chapters are based.

### 2.1 Newton's Method.

Let $g : D \subset R^n \longrightarrow R^n$ be a continuously differentiable mapping and suppose that it is required to find $x^* \in D$ such that

$$g(x^*) = 0 . \tag{2.1.1}$$

Let $\hat{x}$ be an estimate of $x^*$. If we expand $g$ about $\hat{x}$ by Taylor's theorem we shall have

$$g(x^*) = g(\hat{x}) + g'(\hat{x})h + O(\|h\|^2) , \tag{2.1.2}$$

where $g'(\hat{x}) = ( \partial_j g_i (\hat{x}) )_{n \times n}$, $h \in R^n$, and

$$x^* = \hat{x} + h . \tag{2.1.3}$$

When $g$ is a linear function, from (2.1.2), and (2.1.3) we obtain

$$x^* = \hat{x} - g'(\hat{x})^{-1} g(\hat{x}) , \tag{2.1.4}$$

provided that $g'(\hat{x})^{-1}$ exists. Otherwise, when $\hat{x}$ is sufficiently close

to $x^*$, by neglecting the term $0(\|h\|^2)$ in (2.1.2) we obtain a new approximation to $x^*$, namely $\bar{x}$ where

$$\bar{x} = \hat{x} - g'(\hat{x})^{-1} g(\hat{x}). \qquad (2.1.5)$$

This gives rise to Newton's method, which consists of generating the sequence $(x^{(k)})$ from

$$x^{(k+1)} = x^{(k)} - g'(x^{(k)})^{-1} g(x^{(k)}) \quad (k \geq 0) \qquad (2.1.6)$$

with $x^{(0)}$ given. The following results are due to Kantorovich (1964).

Theorem 2.1.1

If 1. $g : S \subset R^n \longrightarrow R^n$ is given mapping and $g \in C^2(S)$, where, $S = B[x^{(0)}, r]$ for some $r > 0$ ;

2. $g'(x^{(0)})$ exists and $\exists K_1$ such that

$$\|g'(x^{(0)})^{-1}\| \leq K_1 ;$$

3. $\exists K_2 > 0$ such that

$$\|g''(x)\| \leq K_2 \qquad (\forall x \in S) ;$$

4. $\exists K_3 > 0$ such that

$$\|g'(x^{(0)})^{-1} g(x^{(0)})\| \leq K_3 ;$$

5. $K = K_1 K_2 K_3 \leqslant 1/2$;

6. $(1 - \sqrt{1-2K}\,)\, K_3 \;/\, K \leqslant r$,

then the Newton sequence generated from (2.1.6) converges to a solution $x^*$ of $g(x) = 0$ which exists in $S$, and

$$\|x^{(K)} - x^*\| \leqslant (1/2)^{K-1} \cdot (2K)^{2^K - 1} K_3 \qquad (\forall k \geqslant 0)$$

Proof.

See Rall (1969) or Kantorovich and Akilov (1964). □

## 2.2 The Extended Iterative Methods

Traub (1964) and Bosarge and Falb (1969), (1970) among others have described a higher order iterative procedure for the solution of $g(x) = 0$, where $g : R^n \longrightarrow R^n$ is a given $F$ - differentiable mapping. The results of Bosarge and Falb apply to $g : X \longrightarrow X$ where $X$ is an arbitrary Banach space. Wolfe (1978a) has considered an iterative method $M_0$, characterized by

$$x^{(K+1)} = G_0(x^{(K)}) \quad (\forall K \geqslant 0), \quad (x^{(0)} \text{ prescribed}) \qquad (2.2.1)$$

for the solution of the operator equation $g(x) = 0$, where

$G_0 : X \longrightarrow X$, and $g : X \longrightarrow X$ are given operators and $X$ is a

Banach space, and has constructed a family of methods $M_p$ $(p \geqslant 1)$

characterized by

$$x^{(k+1)} = G_p(x^{(k)}) \quad (\forall k \geq 0) \ (x^{(0)} \text{ prescribed}), \quad (2.2.2)$$

where $G_p : X \longrightarrow X$ is defined recursively by

$$G_i(x) = G_0(x) - \sum_{j=1}^{i} g'(w(x))^{-1} g(G_{j-1}(x)) \quad (1 \leq i \leq p), \quad (2.2.3)$$

in which $w : X \longrightarrow X$ is a given operator. The iterative method $M_p$ $(p \geq 1)$ has been referred to as an extension of the iterative method $M_0$.

The convergence of the sequence generated from (2.2.2) and (2.2.3) is guaranteed by the following theorem.

Theorem 2.2.1

If 1. $P : X \longrightarrow X$ is a given operator and X is a Banach space;

2. $\exists\, x^* \in X$ such that $x^* = P(x^*)$;

3. $P \in C^2(S)$, where $S = B[x^*, r]$ for some $r > 0$;

4. $(I - P'(x))^{-1}$ exists $(\forall x \in S)$, and for some $B > 0$,

$$\sup_{x \in S} \| (I - P'(x))^{-1} \| \leq B;$$

5. $P'' : X \longrightarrow L(X, L(X))$ is such that for some $K > 0$,

$$\sup_{x \in S} \| P''(x) \| \leq K;$$

6. $x^{(0)} \in S$ ;

7. $w : X \longrightarrow X$ is such that for some $a > 0$, and $\mu \geqslant 1$,

$$\| w(x) - x^* \| \leqslant a \| x - x^* \|^{\mu} , \quad (\forall x \in S) ;$$

8. $G_0 : X \longrightarrow X$, is such that for some $b > 0$ and $\nu \geqslant \mu$ ,

$$\| G_0(x) - x^* \| \leqslant b \| x - x^* \|^{\nu} \quad (\forall x \in S) ;$$

9. $BKr < 2/5$ ;

10. $ar^{\mu - 1} \leqslant 1$ ;

11. $br^{\nu - 1} \leqslant 1$,

then the sequence $(x^{(k)})$ generated from (2.2.2) with $G_p$ ( $p \geqslant 1$) defined by (2.2.3) lies in S and converges to $x^*$ with order of convergence at least $\nu + p\mu$ . Moreover, the rate of convergence is given by

$$\| x^{(k+1)} - x^* \| \leqslant C_p \| x^{(k)} - x^* \|^{\nu + p\mu} \quad (\forall k \geqslant 0), \qquad (2.2.4)$$

where $C_p$ is defined recursively by

$$C_0 = b , \qquad \text{and}$$

$$C_i = BKC_{i-1} \left[ 2a + 3C_{i-1} r^{\nu + (i-2)\mu} \right] / 2 \quad (i \geqslant 1). \qquad (2.2.5)$$

Proof.

See Wolfe (1978a). □

The existence and uniqueness of $x^*$ such that $x^* = P(x^*)$ is guaranteed by a theorem given by Wolfe (1978a).

Different choices of $G_0 : X \longrightarrow X$ in (2.2.1) and $w : X \longrightarrow X$ in (2.2.3) give rise to different members of the class $M_P$. For example, if we define $G_0$ and w according to

$$G_0(x) = x - g'(x)^{-1} g(x) \qquad (\forall\, x \in X), \tag{2.2.6}$$

and

$$w(x) = x \qquad (\forall x \in X), \tag{2.2.7}$$

which is the Newton's method then the iterative method characterized by (2.2.2) and (2.2.3) with $p \geqslant 1$, the method described by Bosarge and Falb (1969), (1970) is obtained. By Theorem 2.2.1, the iterative method corresponding to (2.2.2), (2.2.3), (2.2.6) and (2.2.7) has order of convergence $p + 2 (\forall p \geqslant 1)$. The idea underlying the Extended Newton Method has been used by Brent (1973) for solving systems of nonlinear algebraic equations.

In particular, Wolfe (1978a) has described the iterative methods M 2.2.1 and M 2.2.2, both of which have order of convergence $5 + 2p$. Iterative method M 2.2.1 consists of generating the sequence $(x^{(k)})$ from (2.2.2) and (2.2.3) with $x^{(0)}$ given where w and $G_0$ are defined by

$$w(x) = y - \tfrac{1}{2} g'(x)^{-1} g(y), \tag{2.2.8}$$

where

$$y = x - g'(x)^{-1} g(x), \tag{2.2.9}$$

and $$G_0(x) = y - g'(w(x))^{-1} g(y) . \qquad (2.2.10)$$

Iterative method M 2.2.2 corresponds to

$$w(x) = x - g'(x)^{-1} g(x) , \qquad (2.2.11)$$

and

$$G_0(x) = y - g'(w(x))^{-1} g(y) , \qquad (2.2.12)$$

where

$$y = x - 2 (g'(x) + g'(w(x))^{-1} g(x) . \qquad (2.2.13)$$

The following theorem contains another higher order method for solving $g(x) = 0$. This method appears not to have been used previously.

Theorem 2.2.2

If 1. $g : D \subset X \longrightarrow X$ is given operator on

a Banach space X and $\exists x^*$, such that $g(x^*) = 0$;

2. $g \in C^3(S)$, where $S = B[x^*, r] \subset D$ for some $r > 0$ ;

3. $g'(x)^{-1}$ exists ( $\forall x \in S$ ) and $\exists B >$ such that

$$\sup_{x \in S} \|g'(x)^{-1}\| \leqslant B ;$$

4. $\exists D$ such that

$$\sup_{x \in S} \| g'(x) \| \leqslant D ;$$

5. $\exists K$ such that

$$\sup_{x \in S} \| g''(x) \| \leqslant K ;$$

6. $\exists L$ such that

$$\sup_{x \in S} \| g'''(x) \| \leqslant L ;$$

7. $BKr < 1$ ;

8. $\frac{(BKr)}{42} \left[ 2BLr^2 + 81BD + 15 \right] < 1$ ;

9. $x^{(0)} \in S$ ,

then the sequences $(x^{(K)})$, $(y^{(K)})$ and $(z^{(K)})$ generated from

$$y^{(K)} = x^{(K)} - g'(x^{(K)})^{-1} g(x^{(K)}) \quad (k \geqslant 0); \qquad (2.2.14)$$

$$z^{(K)} = y^{(K)} - g'(x^{(K)})^{-1} g(y^{(K)}) \quad (k \geqslant 0); \qquad (2.2.15)$$

$$x^{(K+1)} = y^{(K)} - g'(z^{(K)})^{-1} g(y^{(K)}) \quad (k \geqslant 0); \qquad (2.2.16)$$

remain in $S$ and converges to $x^*$ and

$$\| y^{(k)} - x^* \| \leqslant \frac{BK}{2} \| x^{(k)} - x^* \|^2 \quad (k \geqslant 0) ; \tag{2.2.17}$$

$$\| z^{(k)} - x^* \| \leqslant \frac{5}{8} (BK)^2 \| x^{(k)} - x^* \|^3 \quad (k \geqslant 0) ; \tag{2.2.18}$$

$$\| x^{(k+1)} - x^* \| \leqslant C \| x^{(k)} - x^* \|^4 \quad (k \geqslant 0) ; \tag{2.2.19}$$

where

$$C = \frac{B (BK)^3}{42} \left[ (2Lr + 15K)r + 162D \right] . \tag{2.2.20}$$

Proof

It is easily verified that

$$\| y^{(0)} - x^* \| \leqslant \frac{BK}{2} \| x^{(0)} - x^* \|^2 . \tag{2.2.21}$$

Since $BKr < 1$ , then from (2.2.21), $y^{(0)} \in S$ . Also, we have

$$\| z^{(0)} - x^* \| \leqslant \| g'(x^{(0)})^{-1} \| \left\{ \| g'(x^{(0)}) - g'(x^*) \| \| y^{(0)} - x^* \| + \right.$$

$$\left. \| g'(x^*)(y^{(0)} - x^*) - g(y^{(0)}) + g(x^*) \| \right\} \tag{2.2.22}$$

By hypotheses 3, 5 and the Mean-Value theorem, from (2.2.21) and (2.2.22) we obtain

$$\| z^{(0)} - x^* \| \leqslant \frac{5}{8} (BK)^2 \| x^{(0)} - x^* \|^3 , \tag{2.2.23}$$

so that by Hypothesis 7, $z^{(0)} \in S$ , and then by Hypothesis 3,

$$\| g'(z^{(0)})^{-1} \| \leq B . \tag{2.2.24}$$

Let $h = z^{(0)} - y^{(0)}$ . Then we have

$$\| \tfrac{1}{2} g''(y^{(0)})h \| \leq \frac{K}{2} \left[ \| z^{(0)} - x^{*} \| + \| y^{(0)} - x^{*} \| \right]$$

$$\leq \frac{9}{16} BK^{2} \| x^{(0)} - x^{*} \|^{2} \tag{2.2.25}$$

and

$$\| g'(y^{(0)})^{-1} \| \cdot \| \tfrac{1}{2} g''(y^{(0)})h \| < \frac{9}{16} (BKr)^{2} \tag{2.2.26}$$

$$< \frac{9}{16} .$$

Let $A = g'(y^{(0)}) + \tfrac{1}{2} g''(y^{(0)})h$ . (2.2.27)

Then, from (2.2.25) and hypotheses 3, and 7 we have

$$\| g'(y^{(0)})^{-1} \| \cdot \| g'(y^{(0)}) - A \| \leq \frac{9}{16} (BKr)^{2}$$

$$< 1 ,$$

so, by Banach's Lemma $A^{-1}$ exists and

$$\| A^{-1} \| \leq \frac{16}{7} B . \tag{2.2.28}$$

Therefore,

$$\| x^{(1)} - x^{*} \| \leq \| A^{-1} \| \quad \| A (y^{(0)} - x^{*}) - g(y^{(0)}) + g(x^{*}) \|$$

$$+ \| A^{-1} \| \quad \| g'(z^{(0)})^{-1} \| \quad \| g'(z^{(0)}) - A \| D \| y^{(0)} - x^{*} \| . \tag{2.2.29}$$

Also by (2.2.21), (2.2.23), and hypotheses 5 and 6 we obtain

$$\| A\,(y^{(0)} - x^{*}) - g\,(y^{(0)}) + g\,(x^{*}) \| \leq \frac{(BK)^3}{8}\left[\frac{Lr}{6} + \frac{5}{4}K\right] \| x^{(0)} - x^{*} \|^5 , \tag{2.2.30}$$

and

$$\| A^{-1} \| \quad \| g'(z^{(0)})^{-1} \| \quad \| g'(z^{(0)}) - A \| \; D \, \| y^{(0)} - x^{*} \| \leq \frac{27 B^4 K^3 D}{7} \| x^{(0)} - x^{*} \|^4 . \tag{2.2.31}$$

Therefore, by (2.2.28) - (2.2.31) we have

$$\| x^{(1)} - x^{*} \| \leq \frac{B^4 K^3}{7}\left[\left(\tfrac{1}{3}Lr + \frac{5K}{2}\right)r + 27D\right] \| x^{(0)} - x^{*} \|^4$$

$$= C \, \| x^{(0)} - x^{*} \|^4 \tag{2.2.32}$$

Since $BKr < 1$ , then by Hypothesis 8, from (2.2.32) we have

$$C r^3 \leq \frac{B^2 K^2}{7}\left[\frac{BLr^2}{3} + 27BD + \frac{5}{2}\right] r^2 \tag{2.2.33}$$

$$< 1,$$

so, $\| x^{(1)} - x^{*} \| \leq r$ whence $x^{(1)} \in S$. Therefore, (2.2.17) - (2.2.19) hold for $k = 0$ . By a similar argument to that which was used in going from $k = 0$ to $k = 1$ it can be proved that (2.2.17) - (2.2.19) hold for $k + 1$ if they hold for $k$. Therefore, by induction on $k$, the theorem is proved. Furthermore we have, by (2.2.19)

$$\| x^{(K+1)} - x^{*} \| \leq (Cr^3)^K \, \| x^{(0)} - x^{*} \| .$$

Thus, by (2.2.33) the sequence $(x^{(K)})$ converges. □

The following theorem is a consequence of theorems (2.2.1) and (2.2.2).

Theorem 2.2.3

If 1. the hypotheses of Theorem 2.2.2 hold;

2. $B\,\frac{(BKr)^3}{42}\left[(2Lr + 15K)r + 162D\right] < 1$ ;

3. $w : X \longrightarrow X$ and $G_0 : X \longrightarrow X$

in (2.2.1) and (2.2.3) are defined according to

$$w(x) = y - g'(x)^{-1}\, g(y)\,, \tag{2.2.34}$$

where

$$y = x - g'(x)^{-1}\, g(x) \tag{2.2.35}$$

and

$$G_0(x) = y - g'(w(x))^{-1}\, g(y)\,, \tag{2.2.36}$$

then the iterative method $M_p$ defined by (2.2.2) and (2.2.3) has order of convergence $4 + 3p$, $(\forall p \geqslant 0)$.

Proof.

By Theorem 2.2.2, we have

$\mu = 3$, $\nu = 4$, $a = 5\,(BK)^2 / 8$ and $b = C$, where $C$ is determined

by (2.2.20). Therefore, by hypotheses 1 and 2, we have $ar^{\mu-1} < 1$ and $br^{\nu-1} < 1$, so, the conclusions of Theorem 2.2.1 hold. □

The iterative method $M_p$ ( $p \geqslant 1$) characterized by Theorem 2.2.3 will be referred to as M 2.2.3.

So far we have quoted some results which show how an iterative method $M_0$ with order of convergence $\nu$ can be extended in such a way that under reasonable conditions, the order of convergence of the new method is higher. Also, in every method which has been considered, it is assumed that the Jacobian is available analytically. But often, in practice, either the Jacobian is difficult to derive analytically or it is expensive to compute. In the next section, we show that the convergence rate for the extended Newton method holds even if the Jacobian is approximated.

## 2.3 Approximated Multipoint Methods.

Let X be a Banach space and $p \geqslant 1$ be a fixed integer. For the given operators $g : X \longrightarrow X$ and $U : X \longrightarrow L(X)$, define the operator $\Psi_p : X \longrightarrow X$ recursively according to

$$\Psi_0(x) = x \quad (\forall x \in X), \tag{2.3.1}$$

$$\Psi_i(x) = x - \sum_{j=1}^{i} U(x)^{-1} g(\Psi_{j-1}(x)) \quad (1 \leqslant i \leqslant p; \forall x \in X), \tag{2.3.2}$$

provided that $U(x)^{-1}$ exists.

From (2.3.2) we have

$$\Psi_{i+1}(x) = \Psi_i(x) - U(x)^{-1} g(\Psi_i(x))$$

$$= [I - U(x)^{-1} g](\Psi_i(x)) \quad (1 \leqslant i \leqslant p). \tag{2.3.3}$$

Thus, for $i = p$,

$$\Psi_{p+1}(x) = [I - U(x)^{-1} g](\Psi_p(x)) \quad (\forall x \in X). \tag{2.3.4}$$

From (2.3.1) - (2.3.4), by mathematical induction we obtain

$$\Psi_p(x) = [I - U(x)^{-1} g]^p(x) \quad (\forall p \geqslant 1; \forall x \in X). \tag{2.3.5}$$

Theorem 2.3.1

If 1. $g : X \longrightarrow X$ is a given operator on a Banach space and

$T : X \longrightarrow X$ is given by $T \stackrel{D}{=} I - g$ ;

2. $W : X \longrightarrow L(X)$ is a given approximation of $T'(x)$ and

$U : X \longrightarrow L(X)$ is given by $U(x) = I - W(x)$ ;

3. $x^* \in X$ is such that $x^* = T(x^*)$ ;

4. $T \in C^2(S)$ , where $S = B[x^*, r]$ for some $r > 0$ ;

5. $(I - W(x))^{-1}$ exists $(\forall x \in S)$, and $\exists B > 0$ such that

$$\sup_{x \in S} \| (I - W(x))^{-1} \| \leqslant B ;$$

6. $\exists K$ such that

$$\sup_{x \in S} \| T''(x) \| \leqslant K ;$$

7. $\exists L$ such that

$$\| W(x) - T'(x) \| \leqslant L \| x - x^* \| \quad (\forall x \in S) ;$$

8. $B(3K + 2L)\, r < 2$ ;

9. $BLr < 1$ ;

10. $x^{(0)} \in S$ ;

11. for a given interger $p \geqslant 1$, the sequence $(x^{(K)})$ is generated from

$$x^{(K+1)} = \Psi_p(x^{(K)}) \quad (k \geqslant 0) , \qquad (2.3.6)$$

where $\Psi_P$ is given by (2.3.1) and (2.3.2), then

(a) the sequence $(x^{(K)})$ converges to $x^*$, and $x^{(K)} \in S \quad (\forall k \geqslant 0)$;

(b) the rate of convergence is given by

$$\| x^{(K+1)} - x^* \| \leqslant C_P \| x^{(K)} - x^* \|^{P+1} \quad (\forall k \geqslant 0), \qquad (2.3.7)$$

where

$$C_1 = B(K + 2L)/2, \qquad (2.3.8)$$

and

$$C_{P+1} = BC_P \left[ K + L + KC_P r^P /2 \right] \quad (\forall p \geqslant 1) \qquad (2.3.9)$$

Proof.

By hypotheses 5, and 7 we have

$$\| (I - W(x))^{-1} \| \| (I - T'(x)) - (I - W(x)) \| \leqslant BL \| x - x^* \|$$

$$\leqslant BLr \quad (\forall x \in S).$$

So, by Banach's lemma and Hypothesis 9, $(I - T'(x))^{-1}$ exists $(\forall x \in S)$ and

$$\| (I - T'(x))^{-1} \| \leqslant B/(1 - BLr).$$

Now, by hypotheses 2, and 1 , and (2.3.5),

$$\Psi_P(x) = \left[ (I - W(x))^{-1} (T - W(x)) \right]^P (x) \quad (\forall x \in S), \qquad (2.3.10).$$

Thus, by hypotheses 5, 6, 7 and 8, and (2.3.6), for p = 1,

$$\| x^{(1)} - x^* \| \leq \| x^* - (I - W(x^{(0)}))^{-1} (T - W(x^{(0)}))(x^{(0)}) \|$$

$$\leq B\,(K/2 + L)\, \| x^{(0)} - x^* \|^2$$

$$= C_1 \| x^{(0)} - x^* \|^2 , \qquad (2.3.11)$$

so that by Hypothesis 8, $C_1 r < 1$ and therefore $x^{(1)} \in S$ and (2.3.7) holds for $k = 0$. By a similar argument to that which was used in going from $k = 0$ to $k = 1$, it can be proved that $x^{(k)} \in S$ $(\forall k \geq 0)$ and (2.3.7) holds. Also, from (2.3.10) by induction on p, (2.3.7) can be proved. Moreover, if we assume that for some $i \geq 1$

$$C_i r^i < 1 \qquad (2.3.12)$$

then by (2.3.9) and Hypothesis 8,

$$C_{i+1} r^{i+1} = BC_i \left[ K + L + KC_i r^i /2 \right] r^{i+1}$$

$$< B \left[ K + L + K/2 \right] r$$

$$< 1 .$$

Since (2.3.12) holds for $i = 1$, therefore by induction

$$C_i r^i < 1 \qquad (i = 1, \ldots, p),$$

and thus, by (2.3.7) the sequence $(x^{(k)})$ converges to $x^*$. □

Corollary 2.3.1

If 1. $g \in C^2(S)$ , where $S = B[x^*, r]$ in which $r > 0$ and $g(x^*) = 0$;

2. $U(x)^{-1}$ exists on $S$ ( $\forall x \in S$ ), and $\exists B > 0$ such that

$$\sup_{x \in S} \| U(x)^{-1} \| \leqslant B ;$$

3. $\exists K > 0$ such that

$$\sup_{x \in S} \| g''(x) \| \leqslant K ;$$

4. $\exists L > 0$ such that

$$\| U(x) - g''(x) \| \leqslant L \| x - x^* \| \quad (\forall x \in S) ;$$

5. $B(3K + 2L)r / 2 < 1$;

6. $BLr < 1$;

7. $x^{(0)} \in S$ ;

8. Hypothesis 11 of Theorem 2.3.1 holds, then the sequence $(x^{(K)})$ converges to $x^*$ and (2.3.7) – (2.3.9) hold. □

As an application of Corollary 2.3.1, let $X = R^n$ and $g : R^n \longrightarrow R^n$ be defined by

$$g(x) = A(x)^T f(x) ,$$

where $f : R^n \longrightarrow R^m$ $(m \geqslant n)$ is a given

mapping and

$$A(x) = ( \partial_j f_i (x)) \quad (i = 1, 2, \ldots, m; \quad j = 1, 2, \ldots, n).$$

Therefore $g'(x) = A(x)^T A(x) + A'(x)^T f(x)$ .

Now, let $U : R^n \longrightarrow L(R^n)$ be defined by

$$U(x) = A(x)^T A(x) + \mu(x) I_n$$

where $I_n$ is an n x n unit matrix, and $\mu : R^n \longrightarrow R^1$ is a given functional. Then the following theorem is a consequence of Corollary 2.3.1.

Theorem 2.3.2

If 1. $f : R^n \longrightarrow R^m$ is a given mapping and

$$f(x^*) = 0 \quad \text{where } x^* \in R^n ;$$

2. $f \in C^3(S)$ where $S = B[x^*, r]$ for some $r > 0$ ;

3. $\mu(x) = C \|f(x)\|_2 \quad (\forall x \in S)$, where C is a given scalar;

4. $U(x)^{-1}$ exists $(\forall x \in S)$, where

$$U(x) = A(x)^T A(x) + \mu(x) I_n , \quad (\forall x \in S)$$

in which $A(x) = (\partial_j f_i(x))_{m \times n}$ ;

5. $g : R^n \longrightarrow R^n$ is defined by

$$g(x) = A(x)^T f(x) \quad (\forall x \in S) ;$$

6. $\exists B > 0$ such that

$$\sup_{x \in S} \| U(x)^{-1} \| \leq B ;$$

7. $\exists K > 0$ such that

$$\sup_{x \in S} \| g''(x) \| \leqslant K ;$$

8. $\exists L_1 > 0$ such that

$$\| A'(x)^T f(x) - A'(y)^T f(y) \| \leqslant L_1 \| x - y \| \quad (\forall x, y \in S);$$

9. $\exists L_2 > 0$ such that

$$\| F(x) - F(y) \| \leqslant L_2 \| x - y \| \quad (\forall x, y \in S),$$

10. $L = L_1 + CL_2$ ;

11. $B(3K + 2L) r/2 < 1$;

12. $x^{(0)} \in S$;

13. the sequence $(x^{(k)})$ is generated from

$$x^{(k+1)} = \Psi_p(x^{(k)}), \quad (k \geqslant 0) \qquad (2.3.13)$$

where $(1 \leqslant i \leqslant p)$ is defined recursively by

$$\Psi_i(x) = x - \sum_{j=1}^{i} U(x)^{-1} A(\Psi_{j-1}(x))^T f(\Psi_{j-1}(x)) \quad (1 \leqslant i \leqslant p) \qquad (2.3.14)$$

$$\Psi_0(x) = x \quad (\forall x \in R^n), \qquad (2.3.15)$$

then the conclusions (a) and (b) of Theorem 2.3.1 hold. □

In connection with Theorem 2.3.2, the following theorem has been proved by Wolfe (1978b).

Theorem 2.3.3

If 1. the hypotheses 1 - 10 of theorem 2.3.2 hold;

2. $x^{(0)} \in S$

3. $\exists L_3 > 0$ such that

$$\| A(x) - A(y) \| \leqslant L_3 \| x - y \| \quad (\forall x, y \in S) ;$$

4. $\exists L_4 > 0$ such that

$$\| f(x) - f(y) \| \leqslant L_4 \| x - y \| \quad (\forall x, y \in S) ;$$

5. $B \left[ 3K + 2 (L + L_3 L_4) \right] r/2 < 1 ;$

6. the sequence $(x^{(K)})$ is generated from

$$x^{(K+1)} = \Psi_p (x^{(K)}) , \qquad (k \geqslant 0) ,$$

where $\Psi_i$ $(1 \leqslant i \leqslant p)$ is defined recursively by

$$\Psi_i (x) = x - \sum_{j=1}^{i} U(x)^{-1} A(x)^T f ( \Psi_{j-1}(x)) , \quad (1 \leqslant i \leqslant p) \qquad (2.3.16)$$

and

$$\Psi_0 (x) = x \qquad (\forall x \in R^n) ; \qquad (2.3.17)$$

then the sequence $(x^{(K)})$ converges to $x^*$. Furthermore,

$$\| x^{(K+1)} - x^* \| \leqslant C_p \| x^{(K)} - x^* \|^{p+1} \qquad (k \geqslant 0) ,$$

where

$$C_1 = B\,(K + 2L)\,/2\ ,$$

and $$C_p = BC_{p-1}\left[(K + L + L_3\,L_4\,) + (K/2 + L_3\,L_4\,)C_{p-1}\,r^{p-1}\right]$$

$$(p \geqslant 1)\ .\ \square$$

The following theorem is used to construct an iterative method for least squares problem in Chapter 5.

Theorem 2.3.4

If 1. $g : X \longrightarrow Y$ is a given operator, where X and Y are Banach spaces;

2. $g\,(x^*\,) = 0_Y$ ;

3. $g \in C^3\,(S)$ where $S = B\,(x^*\ , r)$ for some $r > 0$ ;

4. $\exists K > 0$ such that

$$\|\,g''\,(x)\,\| \leqslant K \quad (\forall x \in S)\ ;$$

5. $g'\,(x)^{-1}$ exists $(\forall x \in S)$ and $\exists B > 0$

such that $\|\,g'\,(x)^{-1}\,\| \leqslant B \quad (\forall x \in S)$ ;

6. $U : X \longrightarrow L\,(X, Y)$ is such that

$$\|\,U\,(\bar{x}) - g'\,(\bar{x})\,\| \leqslant L\,\|\hat{x} - \bar{x}^*\| \qquad (\forall \hat{x} \in B\,(x^*\ , r)),$$

where

$$\bar{x} = \hat{x} - g'\,(\hat{x})^{-1}\,g\,(\hat{x})\ ;$$

7. $T : X \longrightarrow Y$ is such that $\exists M > 0$ ,

$$\| T(\bar{x}) - g(\bar{x}) \| \leqslant M \| \hat{x} - x^* \|^p \quad (\forall \hat{x} \in B(x^*, r)) ,$$

where $p$ is a positive integer, and $p \leqslant 3$ ;

8. $BLr < 1$ ;

9. $BKr < 1$ ;

10. $B\left[B^2 K^3 r^{4-p} + 4BLKr^{3-p} + 8M\right] r^{p-1} < 8(1 - BLr)$ ;

11. $x^{(0)} \in S$ ;

12. the sequences $(\bar{x}^{(k)})$, and $(x^{(k)})$ are generated from

$$\bar{x}^{(k)} = x^{(k)} - g'(x^{(k)})^{-1} g(x^{(k)}) \quad (\forall k \geqslant 0) \qquad (2.3.18)$$

$$x^{(k+1)} = \bar{x}^{(k)} - U(\bar{x}^{(k)})^{-1} T(\bar{x}^{(k)}) \quad (\forall k \geqslant 0) \qquad (2.3.19)$$

then

(a) the sequences $(x^{(k)})$ and $(\bar{x}^{(k)})$ converge to $x^*$ ;

(b) $$\| \bar{x}^{(k)} - x^* \| \leqslant \frac{BK}{2} \| x^{(k)} - x^* \|^2 \quad (\forall k \geqslant 0) \qquad (2.3.20)$$

(c) $$\| x^{(k+1)} - x^* \| \leqslant C \| x^{(k)} - x^* \|^p \quad (\forall k \geqslant 0) \qquad (2.3.21)$$

where

$$C = B\left[B^2 K^3 r^{4-p} + 4BLKr^{3-p} + 8M\right] / 8(1 - BLr) .$$

Proof

By hypotheses 3, 6 and 12, for $k=0$ we have

$$\| \bar{x}^{(0)} - x^{*} \| \leqslant \frac{BK}{2} \| x^{(0)} - x^{*} \|^{2} .$$

thus, by Hypothesis 9, $\bar{x}^{(0)} \in B(x^{*}, r)$. By hypotheses 6, and 8 and Banach's lemma, $U(\bar{x}^{(0)})^{-1}$ exists and

$$\| U(\bar{x}^{(0)})^{-1} \| \leqslant B/(1 - BLr) .$$

Therefore, by (2.3.19) for $k = 0$ and hypotheses 4, 6, and 7 we have

$$\| x^{(1)} - x^{*} \| \leqslant \| U(\bar{x}^{(0)})^{-1} \| \cdot \| [g'(\bar{x}^{(0)})(\bar{x}^{(0)} - x^{*}) - g(\bar{x}^{(0)}) + g(x^{*})]$$

$$+ (U(\bar{x}^{(0)}) - g'(\bar{x}^{(0)}))(\bar{x}^{(0)} - x^{*}) - (T(\bar{x}^{(0)}) - g(\bar{x}^{(0)})) \|$$

$$\leqslant \frac{B}{(1 - BLr)} \cdot \left[ B^{2} K^{3} r^{4-p} /8 + BKLr^{3-p} /2 + M \right] \| x^{(0)} - x^{*} \|^{p} .$$

so by Hypothesis 10, $x^{(1)} \in B(x^{*}, r)$.

Therefore (b) and (c) hold for $k = 0$ . By a similar argument to that which was used in going from $k = 0$ to $k = 1$ , it can be proved that (b) and (c) hold for $k + 1$ if they hold for $k$ . Therefore by induction on $k$ , (b) and (c) are proved.

From (2.3.21),

$$\| x^{(k+1)} - x^{*} \| \leqslant Cr^{p-1} \| x^{(k)} - x^{*} \| ,$$

where, by hypotheses 8 - 10, $Cr^{p-1} < 1$ . Therefore

$$x^{(k)} \longrightarrow x^{*} \quad (k \longrightarrow \infty) . \quad \square$$

# Chapter 3

## Some Modifications of the Newton-type Algorithms of Gill and Murray.

In this chapter some algorithms for unconstrained minimization based on the iterative methods discussed in Chapter 2 are presented.

### 3.1 The Newton-type Algorithms of Gill and Murray.

If $F : R^n \longrightarrow R^1$ is continuously differentiable then a minimizer $x^*$ of $F$ satisfies

$$g(x) = 0 , \qquad (3.1.1)$$

where $g : R^n \longrightarrow R^n$ is the gradient vector of $F$.
Therefore any of the iterative methods for the solution of (3.1.1) which are discussed in Chapter 2 may, in principle, be used to construct an algorithm for estimating $x^*$ .
Safeguarded algorithms which are based upon Newton's method for solving (3.1.1) have been discussed by Goldstein and Price (1967), Greenstadt (1967), Fiacco and McCormick (1968), Dixon and Biggs (1970), Mathews and Davies (1971), Gill and Murray (1974a), and Fletcher and Freeman (1977). An introductory account of the principal problems which need to be overcome in order to safeguard Newton's method has been given by Wolfe (1978).

The Newton-type algorithm MNA of Gill and Murray (1974a) for estimating an unconstrained minimizer $x^*$ of $F : R^n \longrightarrow R^1$ is as follows:

Algorithm 3.1.1 (MNA)

Suppose that an estimate $x^{(0)}$ of a strong local minimizer $x^*$ of F, and a tolerance $\varepsilon > 0$, are given.

Step 1. Set $k = 0$, and compute $F^{(k)} = F(x^{(k)})$.

Step 2. Compute $g^{(k)} = g(x^{(k)})$.

Step 3. Compute $G^{(k)} = G(x^{(k)})$.

Step 4. Form the modified Cholesky factorization of $G^{(k)}$ such that

$$\bar{G}^{(k)} = L^{(k)} D^{(k)} L^{(k)T}$$

$$= G^{(k)} + E^{(k)}$$

where $L^{(k)}$ is a unit lower triangular matrix, $D^{(k)} = \text{Diag}(d_1^{(k)}, \ldots, d_n^{(k)})$, and $E^{(k)} = \text{Diag}(E_1^{(k)}, \ldots, E_n^{(k)})$. The factorization is such that $\bar{G}^{(k)}$ is positive definite, and $\|E^{(k)}\|_\infty = 0$ if $G^{(k)}$ is positive definite. For details see Gill and Murray (1974a).

Step 5. If $\|g^{(k)}\|_2 \leq \varepsilon$ and $\|E^{(k)}\|_\infty = 0$, then $x^{(k)}$ is regarded as an adequate estimate of $x^*$ and the algorithm is terminated. If $\|g^{(k)}\|_2 > \varepsilon$, then determine $p^{(k)}$ by solving the linear system

$$L^{(k)} D^{(k)} L^{(k)T} p^{(k)} = -g^{(k)}. \qquad (3.1.2)$$

If $\|g^{(k)}\|_2 \leq \varepsilon$ and $\|E^{(k)}\|_\infty \neq 0$, then determine y

by solving the linear system

$$L^{(K)} y = e_j \quad , \tag{3.1.3}$$

where $e_j$ is the column j of the $n \times n$ unit matrix and

$$d_j^{(K)} - E_j^{(K)} = \min_{1 \leqslant i \leqslant n} ( d_i^{(K)} - E_i^{(K)} ) .$$

If $\| g^{(K)} \|_2 = 0$, then set

$$p^{(K)} = y \quad , \tag{3.1.4}$$

If $\| g^{(K)} \|_2 \neq 0$ , then set

$$p^{(K)} = - \text{Sign} (g^{(K)T} y) y \quad . \tag{3.1.5}$$

Step 6. Determine $\alpha^{(K)}$ such that $\alpha^{(K)} \leqslant \lambda$ and

$$F (x^{(K)} + \alpha^{(K)} p^{(K)}) \leqslant F^{(K)}$$

by using Algorithm 1.2.2. During the implementation of the Algorithm 1.2.2, both $F (x^{(K)} + \alpha^{(K)} p^{(K)})$ and $g (x^{(K)} + \alpha^{(K)} p^{(K)})$ are computed.

Step 7. Set $k = k + 1$ and go to Step 3. □

Algorithm 3.1.2 (MNA DIFF) is the same as Algorithm 3.1.1 save that Step 3 of MNADIFF is as follows.

Step 3. Compute the $n \times n$ matrix $Y^{(K)}$ with columns $y_j^{(K)}$ $(j = 1, 2, \ldots, n)$ defined by

$$y_j^{(K)} = (g(x^{(K)} + h_j^{(K)} e_j) - g(x^{(K)}))/h_j^{(K)}, \quad (3.1.6)$$

where $2^{-\frac{t}{2}} \leqslant h_j^{(K)} \leqslant 2^{-\frac{2t}{3}}$ in which $2^{-t}$ is the relative machine precision. Compute the $n \times n$ matrix $G^{(K)}$ from

$$G^{(K)} = (Y^{(K)} + Y^{(K)T})/2 \,. \quad \square \qquad (3.1.7)$$

The following theorems are valid, and their proofs are given in Gill and Murray (1974a).

Theorem 3.1.1.

Let $G^{(K)}$ $(k \geqslant 0)$ be a symmetric matrix with bounded elements. Then the $j^{th}$ diagonal element of the matrix $E^{(K)}$ associated with the modified Cholesky factorization of $G^{(K)}$ is bounded. $\square$

Theorem 3.1.2

If 1. $(G^{(K)})$ is a sequence of symmetric matrices;

2. $\exists \rho$ such that

$$\|G^{(K)}\|_\infty \leqslant \rho \qquad (\forall k \geqslant 0) \,;$$

3. $\bar{G}^{(K)} = L^{(K)} D^{(K)} L^{(K)T} = G^{(K)} + E^{(K)}$,

where, $L^{(K)}$, $D^{(K)}$, and $E^{(K)}$ correspond to the modified

Cholesky factorization of $G^{(k)}$,

then $\exists \nu > 0$ such that

$$x^T \tilde{G}^{(k)} x \geq \nu \| x \|_2^2 \quad (\forall x \in R^n). \quad \square$$

Theorem 3.1.3

If 1. $F : D \subset R^n \longrightarrow R^1$ is given and $F \in C^2(D)$;

2. the set S of critical points of F is finite;

3. $x^{(0)} \in D$ is such that $\bar{\Omega}(F^{(0)})$ is compact and $CO[\bar{\Omega}(F^{(0)})] \subset D$, where $\bar{\Omega}(t)$ denotes the closure of the level set

$$\Omega(t) = \{ x \in D : F(x) \leq t \}$$

and $CO[\bar{\Omega}]$ the closed convex hull of $\bar{\Omega}$ ;

4. $\exists \rho > 0$ such that $\| G(x) \| \leq \rho \quad (\forall x \in \bar{\Omega}(F^{(0)}))$;

5. $\varepsilon = 0$,

then the sequence $(x^{(k)})$ generated from Algorithm 3.1.1 (MNA) or Algorithm 3.1.2 (MNADIFF) is such that

$$x^{(k)} \longrightarrow x^* \quad (\text{as } k \longrightarrow \infty),$$

where $x^* \in S$. $\square$

Theorem 3.1.4

If 1. hypotheses 1 - 4 of Theorem 3.1.3 are valid;

2. $G(x)$ is not positive semi-definite at any point $x \in S$,

then the sequence $(x^{(K)}(\varepsilon))$ generated from Algorithm 3.1.1 is such that

$$\lim_{\varepsilon \to 0} \lim_{K \to \infty} x^{(K)}(\varepsilon) = x^* ,$$

where $x^*$ is a strong local minimizer of F. □

If the sequence $(x^{(K)})$ generated from Algorithm 3.1.1 ultimately lies in a set on which G is uniformly positive definite, then Algorithm 3.1.1 is identical with Newton's method and there is a sequence $(x^{(K)})$ which converges to a strong local minimizer of F. Also there exists an integer $\hat{k}$ such that $\alpha^{(K)} = 1 \quad (\forall k \geq \hat{k})$, and the convergence of $(x^{(K)})$ is superlinear.

As pointed out by Gill and Murray, Algorithms 3.1.1 and 3.1.2 are particularly attractive when G is a band matrix. In this case, Algorithm 3.1.2 would be expected to be superior to quasi-Newton methods. Gill and Murray claim that Algorithm 3.1.2 competes with quasi-Newton methods when G is not sparse and $n \leq 10$, while if G has a known structure such as a fixed band, then Algorithm 3.1.2 is superior to quasi-Newton methods with respect to both function and gradient subroutine calls, storage requirements, and work per iteration.

## 3.2 Extended Newton and Approximated Extended Newton Method.

If $X = R^n$, $P = I - g$, where

$$g = (\partial_1 F, \ldots, \partial_n F)^T ,$$

then the iterative methods defined by (2.2.6), (2.2.7), (2.2.2) and

(2.2.3) with $p = 1$ and $p \geqslant 2$ are Newton and extended Newton methods for locating the critical point of F, when G the Hessian of F is available analytically. Sufficient conditions for the convergence of the sequences generated by these methods are contained in Theorem 2.2.1. Also, when the Hessian G of F is not available analytically, then by Corollary 2.3.1, the approximated Newton iteration and the extended Newton iteration generated from (2.3.1), (2.3.2), and (2.3.6) with $p = 1$ and $p \geqslant 2$ respectively converge to a critical point of F with order of convergence $p + 1$. Therefore, if $x^{(0)}$ is sufficiently close to a critical point $x^*$ of F and G is positive definite at all points in a neighbourhood of $x^*$, then the sequence $(x^{(K)})$ generated by the extended Newton and approximated extended Newton methods will converge to $x^*$ more rapidly than the sequence generated from the Newton (Algorithm 3.1.1) and approximated Newton methods (Algorithm 3.1.2), respectively. Furthermore, at points far removed from $x^*$, it should be possible to economize on the number of evaluations and inversions of G if the extended Newton and approximated Newton methods are used. It is clear, however, that just a Newton (or approximated Newton) method will, in general, fail unless $x^{(0)}$ is sufficiently close to $x^*$, and the extended Newton (or approximated extended Newton) method alone will fail. There is no guarantee that, at points x far removed from $x^*$, $G(x)$ is positive definite; neither is there any guarantee that $F(x^{(K+1)}) < F(x^{(K)})$ if $x^{(K+1)}$ is computed from (2.2.6), (2.2.7), (2.2.2) and (2.2.3) (or (2.3.1), (2.3.2) and (2.3.6) ) with $p \geqslant 2$. The following algorithms, however, will be shown to converge to a critical point of F under the hypotheses of Theorem 3.1.3.

Algorithm 3.2.1 (Extended Newton Method)

Suppose that $x^{(0)}$, p, and $\varepsilon_i > 0$ $(i = 1, 2, 3)$ are given, $p \geqslant 2$.

Step 1. Set $k = 0$ and compute $F^{(K)} = F(x^{(K)})$.

Step 2. Compute $g^{(K)} = g(x^{(K)})$.

Step 3. Compute $G^{(K)} = G(x^{(K)})$.

Step 4. Form the modified Cholesky factorization of $G^{(K)}$ as in Algorithm 3.1.1 $(\bar{G}^{(K)} = L^{(K)} D^{(K)} L^{(K)T} = G^{(K)} + E^{(K)})$.

Step 5. If $\| g^{(K)} \|_2 > \varepsilon_1$, then go to Step 8.

Step 6. If $\| E^{(K)} \|_\infty = 0$, then $x^{(K)}$ is regarded as an adequate estimate of a strong local minimizer $x^*$ of F, and the algorithm is terminated.

Step 7. Determine $p^{(K)}$ by using (3.1.3) - (3.1.5), set $j = 0$, and go to Step 23.

Step 8. Determine $p^{(K)}$ from (3.1.2).

Step 9. If $\min_{1 \leqslant i \leqslant n} \{ d_i^{(K)} \} < \varepsilon_2$ then go to Step 23.

Step 10. If $\| p^{(K)} \|_2 \geqslant \varepsilon_3$, then go to Step 23.

Step 11. Set $j = 1$, $g_0^{(K)} = g^{(K)}$, $p_0^{(K)} = p^{(K)}$, $x_0^{(K)} = x^{(K)}$, $F_0^{(K)} = F^{(K)}$.

Step 12. Determine $x_1^{(K)}$, $F_1^{(K)}$, from

$$x_1^{(K)} = x_0^{(K)} + p_0^{(K)},$$

and

$$F_1^{(K)} = F\left(x_1^{(K)}\right).$$

Step 13. If

$$F_1^{(K)} > F_0^{(K)} + \mu\, g_0^{(K)T} p_0^{(K)},$$

where $\mu \in (0, \frac{1}{2}]$, then set $j = 0$ and go to Step 23.

Step 14. Compute $g_j^{(K)} = g\left(x_j^{(K)}\right)$.

Step 15. If $j = p$ then go to Step 22.

Step 16. If $g_j^{(K)T} p_0^{(K)} \geqslant 0$ then go to Step 22.

Step 17. Compute $p_j^{(K)} = -\bar{G}^{(K)-1} g_j^{(K)}$.

Step 18. If $\| p_j^{(K)} \| \geqslant \varepsilon_3$ then go to Step 22.

Step 19. Determine $x_{j+1}^{(K)}$, $F_{j+1}^{(K)}$ from

$$x_{j+1}^{(K)} = x_j^{(K)} + p_j^{(K)},$$

$$F_{j+1}^{(K)} = F\left(x_{j+1}^{(K)}\right).$$

Step 20. If $F_{j+1}^{(K)} > F_j^{(K)} + \mu\, g_0^{(K)T} p_j^{(K)}$, then go to Step 22.

Step 21. Set $j = j + 1$ and go to Step 14.

Step 22. Set $x^{(k+1)} = x_j^{(k)}$, $F^{(k+1)} = F_j^{(k)}$

$g^{(k+1)} = g_j^{(k)}$, $k = k + 1$, $j = 0$, and go to Step 3.

Step 23. Determine $x^{(k+1)}$, $F^{(k+1)}$, and $g^{(k+1)}$ from

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)},$$

$$F^{(k+1)} = F(x^{(k+1)}),$$

$$g^{(k+1)} = g(x^{(k+1)}),$$

by using Algorithm 1.2.2.

Step 24. Set $k = k + 1$ and go to Step 3. □

Algorithm 3.2.2 is the same as Algorithm 3.2.1 save that Step 3 of Algorithm 3.2.2 is identical with Step 3 of Algorithm 3.1.2.

Steps 1 - 8 of Algorithm 3.2.1 are identical with steps 1 - 5 of Algorithm 3.1.1. Step 9 contains a test which prevents excessively large steps $p^{(k)}$ which could lead to overflow, especially when the objective function contains exponentials. Step 10 contains a more direct check on the step-length. Steps 14 - 21 contain an inner iteration which is indexed by j, where $1 \leq j \leq \mathcal{P}$. The inner iteration permits the same inverse Hessian $G^{(k)-1}$ to be used for up to $\mathcal{P}$ times. The inner iteration is left either because $j = \mathcal{P}$ or because one of the tests in 13, 16, 18, and 20 indicates that the inner iteration should be discontinued.

In algorithms 3.1.1 and 3.1.2 the sequence $(x^{(K)})$ is computed essentially from

$$x^{(K+1)} = x^{(K)} + \alpha^{(K)} p^{(K)}, \tag{3.2.1}$$

where

$$p^{(K)} = -\bar{G}^{(K)-1} g^{(K)}, \tag{3.2.2}$$

and $\alpha^{(K)}$ is such that

$$F^{(K+1)} \leqslant F^{(K)} + \mu \alpha^{(K)} g^{(K)T} p^{(K)}. \tag{3.2.3}$$

In Algorithm 3.2.1, $(x^{(K)})$ is computed essentially from

$$x^{(K+1)} = x^{(K)} - \bar{G}^{(K)-1} \sum_{j=0}^{P_K} g_j^{(K)}, \tag{3.2.4}$$

where $P_K \leqslant P$ varies from iteration to iteration, and is such that

$$g_j^{(K)T} p_0^{(K)} < 0 \quad (j = 0, \ldots, P_K), \tag{3.2.5}$$

and

$$F^{(K+1)} = F^{(K)}_{P_K}$$

$$\leqslant F^{(K)} + \mu g^{(K)T} p^{(K)}, \tag{3.2.6}$$

where

$$p^{(K)} = \sum_{j=0}^{P_K} p_j^{(K)} \tag{3.2.7}$$

Therefore, in Algorithm 3.2.1, $(\forall k \geq 0)$,

$$g^{(k)T} p^{(k)} = \sum_{j=0}^{p_k} g^{(k)T} \bar{G}^{(k)-1} g_j^{(k)}$$

$$= \sum_{j=0}^{p_k} g_j^{(k)T} p_0^{(k)} \qquad (3.2.8)$$

$$< 0 .$$

By inspection of Algorithm 3.2.1, we see that the sequence $(x^{(k)})$ is generated from (3.2.1) with $p^{(k)}$ given by (3.2.7) for some $p_k \leq p$, or with $p^{(k)}$ determined from (3.1.3) — (3.1.5) if $\|g^{(k)}\|_2 \leq \varepsilon_k$ and $G^{(k)}$ is not positive definite. Furthermore, if one of the algorithms 1.2.2 or 1.2.3 is used, then (3.2.3) holds and $\alpha^{(k)} \leq \lambda$ $(\forall k \geq 0)$ for some $\lambda > 0$.

The following theorem guarantees that under the hypotheses of Theorem 3.1.3, the sequence $(x^{(k)})$ generated from Algorithm 3.2.1 converges to a critical point of F.

Theorem 3.2.1

If 1. the hypotheses of Theorem 3.1.3 hold;

2. $\varepsilon = \varepsilon_1$ ,

then the sequence $(x^{(k)})$ generated from Algorithm 3.2.1 or Algorithm 3.2.2 is such that

$$\lim_{k \to \infty} x^{(k)} = x^* ,$$

where $x^* \in S$.

Proof.

We may suppose that $g^{(0)} \neq 0$. Then by (3.2.8)

$$g^{(k)T} p^{(k)} \leqslant 0 \qquad (\forall k \geqslant 0),$$

with equality only if $g^{(k)} = 0$.

By Theorem 1.2.7, either

$$F^{(k)} - F^{(k+1)} \geqslant \sigma(- g^{(k)T} p^{(k)} / \| p^{(k)} \| ) \tag{3.2.9}$$

where $\sigma : R^1 \longrightarrow R^1$ is a forcing function, or

$$F^{(k)} - F^{(k+1)} \geqslant - \mu \lambda g^{(k)T} p^{(k)} \tag{3.2.10}$$

where $\mu \in (0, \frac{1}{2}]$ . Whichever of (3.2.9) or (3.2.10) holds, $(F^{(k)})$ is monotone decreasing sequence and $x^{(k)} \in \bar{\Omega}(F^{(0)})$ $(\forall k \geqslant 0)$. Since $\bar{\Omega}(F^{(0)})$ is compact, $(F^{(k)})$ converges, whence

$$\lim_{k \to \infty} (F^{(k)} - F^{(k+1)}) = 0. \tag{3.2.11}$$

By Algorithm 3.2.1,

$$p^{(k)} = - \bar{G}^{(k)-1} \sum_{j=0}^{p_k} g_j^{(k)}, \tag{3.2.12}$$

and by Theorem 3.1.2, there exists $\nu$ such that

$$g_0^{(k)T} \bar{G}^{(k)-1} g_0^{(k)} \geqslant \nu \| p_0^{(k)} \|^2 \quad (\forall k \geqslant 0). \tag{3.2.13}$$

Therefore by (3.2.12), (3.2.13) and (3.2.5),

$$
\begin{aligned}
-g^{(K)T} p^{(K)} &= \sum_{j=0}^{p_K} g_0^{(K)T} \bar{G}^{(K)-1} g_j^{(K)} \\
&\geqslant \nu \|p_0^{(K)}\|^2 - \sum_{j=1}^{p_K} g_j^{(K)T} p_0^{(K)} \\
&\geqslant \nu \|p_0^{(K)}\|^2 \quad . \qquad (3.2.14)
\end{aligned}
$$

Also, by Theorem 3.1.1, and Hypothesis 4 of Theorem 3.1.3, $\exists\, \bar{\rho}$ such that

$$
\|\bar{G}^{(K)}\| \leqslant \bar{\rho} \qquad (\forall k \geqslant 0) \, .
$$

Therefore

$$
\|g^{(K)}\| \leqslant \bar{\rho} \, \|p_0^{(K)}\| \quad (\forall k \geqslant 0) \, , \qquad (3.2.15)
$$

whence by (3.2.14) and (3.2.7),

$$
\begin{aligned}
-g^{(K)T} p^{(K)} / \|p^{(K)}\| &\geqslant \nu \|p_0^{(K)}\|^2 / \|p^{(K)}\| \\
&\geqslant \nu \|p_0^{(K)}\|^2 / \Big( \|p_0^{(K)}\| + \sum_{j=1}^{p_K} \|p_j^{(K)}\| \Big) \, . \qquad (3.2.16)
\end{aligned}
$$

If (3.2.10) holds, then by (3.2.11), and (3.2.14)

$$
p_0^{(K)} \longrightarrow 0 \ (k \longrightarrow \infty) \, .
$$

Assume that (3.2.9) holds. Then by (3.2.11) and Theorem 1.2.3 we have

$$
\lim_{K \to \infty} \Big( -g^{(K)T} p^{(K)} / \|p^{(K)}\| \Big) = 0 \, . \qquad (3.2.17)
$$

By inspection of Algorithm 3.2.1,(or Algorithm 3.2.2) we see that $F_j^{(k)} < F_{j-1}^{(k)} (j = 1, 2, \ldots, p_k)$, so $x_j^{(k)} \in \Omega(F^{(0)})$ $(j = 0, 1, 2, \ldots, p_k)$. Therefore $p_j^{(k)}$ is bounded $(0 \leqslant j \leqslant p_k)$ $(\forall k \geqslant 0)$. Therefore by (3.2.17) and (3.2.16), $p_0^{(k)} \longrightarrow 0$ as $k \longrightarrow \infty$. Thus, by (3.2.15), $\|g^{(k)}\| \longrightarrow 0$ as $k \longrightarrow \infty$.

Now

$$\|x^{(k+1)} - x^{(k)}\| = \alpha^{(k)} \|p^{(k)}\|$$

where either $\alpha^{(k)} = 1$ or $\alpha^{(k)}$ is determined by one of the algorithms 1.2.2 or 1.2.3. In either case $\alpha^{(k)} \leqslant \Lambda$ $(\forall k \geqslant 0)$ for some $\Lambda > 0$. Therefore

$$\|x^{(k+1)} - x^{(k)}\| \leqslant \Lambda \|p^{(k)}\|$$

$$\leqslant \Lambda \bar{\rho} \sum_{j=0}^{p_k} \|g_j^{(k)}\| . \qquad (3.2.18)$$

Also by hypotheses 1, and 4 of Theorem 3.1.3

$$\|g_1^{(k)} - g_0^{(k)}\| \leqslant \sup_{0 \leqslant \theta \leqslant 1} \|G(x_0^{(k)} + \theta p_0^{(k)})\| \, \|x_1^{(k)} - x_0^{(k)}\|$$

$$\leqslant \rho \|x_1^{(k)} - x_0^{(k)}\|$$

$$\leqslant \rho \|\bar{G}^{(k)-1}\| \, \|g_0^{(k)}\| .$$

Therefore $\|g_1^{(k)}\| \longrightarrow 0$ as $k \longrightarrow \infty$. Similarly, $\|g_j^{(k)}\| \longrightarrow 0$ as $k \longrightarrow \infty$ for any $j$ $(1 \leqslant j \leqslant p_k)$. Therefore by (3.2.18), since $p_k \leqslant p$ $(\forall k \geqslant 0)$, $\|x^{(k+1)} - x^{(k)}\| \longrightarrow 0$ as $k \longrightarrow \infty$. The theorem now follows from Theorem 14.15 of Ortega and Rheinboldt (1970). □

The following theorem may be proved in the same way as Theorem 3.2.1.

Theorem 3.2.2

If 1. the hypotheses of Theorem 3.2.1 are valid;

2. $\varepsilon = \varepsilon_1$ ,

then the sequence $(x^{(K)}(\varepsilon))$ generated from Algorithm 3.2.1 is such that

$$\lim_{\varepsilon \to 0} \lim_{K \to \infty} x^{(K)}(\varepsilon) = x^* ,$$

where $x^*$ is a strong local minimizer of F. □

In Algorithm 3.2.1 (or Algorithm 3.2.2), it is necessary to compute $F_j^{(K)}$ $(j = 0, 1, \ldots, P_K)$ $(\forall k \geqslant 0)$ in order to determine whether to accept $p_j^{(K)}$ for addition to the step which is to be taken from $x^{(K)}$. It is reasonable to conjecture that it may be more efficient to determine the total step $p^{(K)}$ without requiring that

$$F_{j+1}^{(K)} \leqslant F_j^{(K)} + \mu\, g_o^{(K)T}\, p_j^{(K)} \quad (j = 0, 1, \ldots, p_K).$$

If we still require that

$$g_j^{(K)T}\, p_o^{(K)} < 0 \quad (j = 0, 1, \ldots, p_K),$$

then (3.2.8) still holds, and we may therefore still use $p^{(K)}$ in algorithms 1.2.2 and 1.2.3. These considerations give rise to the algorithms 3.2.3 and 3.2.4.

Algorithm 3.2.3

Suppose that $x^{(0)}$, p, and $\varepsilon_i > 0$ $(i = 1, 2, 3)$ are given.

Step 1. Set $k = 0$ and compute $F^{(K)} = F(x^{(K)})$.

Step 2. Compute $g^{(K)} = g(x^{(K)})$.

Step 3. Compute $G^{(K)} = G(x^{(K)})$.

Step 4. Form the modified Cholesky factorization of $G^{(K)}$ as in Algorithm 3.1.1.

Step 5. If $\|g^{(K)}\|_2 \geq \varepsilon_1$ then go to Step 8.

Step 6. If $\|E^{(K)}\|_\infty = 0$, then $x^{(K)}$ is regarded as an adequate estimate of a strong local minimizer of F and the algorithm is terminated.

Step 7. Determine $p^{(K)}$ by using (3.1.3) - (3.1.5), set $j = 0$, and go to Step 22.

Step 8. Determine $p^{(K)}$ from (3.1.2).

Step 9. If $\min_{1 \leq i \leq n} \{ d_i^{(K)} \} < \varepsilon_2$, then go to Step 21.

Step 10. If $\|p^{(K)}\|_2 \geq \varepsilon_3$, then go to Step 21.

Step 11. Set $j = 0$, $g_0^{(K)} = g^{(K)}$, $p_0^{(K)} = p^{(K)}$, $x_0^{(K)} = x^{(K)}$.

Step 12. If $\|p^{(K)}\|_2 > 1$, then go to Step 21.

Step 13. If $j = P$, then go to Step 24.

Step 14. Determine $x_{j+1}^{(K)}$ from

$$x_{j+1}^{(K)} = x_j^{(K)} + p_j^{(K)}.$$

Step 15. Set $j = j + 1$.

Step 16. Compute $g_j^{(K)} = g\left(x_j^{(K)}\right)$.

Step 17. If $g_j^{(K)T} p^{(K)} \geqslant 0$ then set

$$p^{(K)} = \sum_{i=0}^{j-1} p_i^{(K)}$$

and go to Step 21.

Step 18. Compute $p_j^{(K)} = -\bar{G}^{(K)-1} g_j^{(K)}$.

Step 19. If $\|p_j^{(K)}\|_2 \geqslant 1$, then set

$$p^{(K)} = \sum_{i=0}^{j-j_1} p_i^{(K)}$$

and go to Step 21.

Step 20. Go to Step 13.

Step 21. Set $j = 0$.

Step 22. Determine $x^{(K+1)}$, $F^{(K+1)}$, and $g^{(K+1)}$ from

$$x^{(K+1)} = x^{(K)} + \alpha^{(K)} p^{(K)},$$

$$F^{(K+1)} = F\left(x^{(K+1)}\right),$$

and

$$g^{(K+1)} = g\left(x^{(K+1)}\right),$$

by using algorithms 1.2.2 or 1.2.3 as in Algorithm 3.1.1.

Step 23. Set $k = k + 1$ and go to Step 3.

Step 24. Set $p^{(K)} = \sum_{i=0}^{p} p_i^{(K)}$,

and go to Step 21. □

Algorithm 3.2.4 is the same as Algorithm 3.2.3 save that Step 3 of Algorithm 3.2.4 is identical with Step 3 of Algorithm 3.2.2.

Analogues of theorems 3.2.1 and 3.2.2 may be proved for the algorithms 3.2.3 and 3.2.4.

A significant part of the computational labour required to implement the algorithms 3.2.1 - 3.2.4 are that required to evaluate G. At the points x far removed from the minimizer $x^*$, an accurate estimate of G (x) is not required, and it may be conjectured that a quasi-Newton updating formula could be used. At the points near $x^*$, where g (x) is small, an accurate estimate of G (x) would be required in order to take advantage of the local convergence properties of the iteration corresponding to the extended Newton or approximated extended Newton Method. These observations give rise to the following modifications of Algorithm 3.2.2 to give Algorithm 3.2.5.

(a) Replace Step 1 of Algorithm 3.2.2 with

Step 1′. Set $k = 0$, $n_2 = 0$, and compute $F^{(k)} = F\left(x^{(k)}\right)$.

(b) Replace Step 3 of Algorithm 3.2.2 with

Step $3'.1$. Set $n_1 = 2$.

Step $3'.2$. If $\|g^{(K)}\| \geqslant \varepsilon_4$ and $n_2 = 0$ then set $n_1 = 1$.

Step $3'.3$. If $n_1 = 2$ then compute $G^{(K)}$ as in Algorithm 3.2.2.

Step $3'.4$. If $n_1 = 1$ then compute $G^{(K)}$ from (1.5.3), noting that for $k = 0$, we set $G^{(K)} = I_n$, where $I_n$ is an $n \times n$ unit matrix.

(c) Insert steps $4'$ and $4''$ between steps 4 and 5 of Algorithm 3.2.2.

Step $4'$. Set $n_2 = 0$.

Step $4''$. If $\|E^{(K)}\|_\infty \neq 0$ and $n_1 = 2$ then set $n_2 = 1$.

(d) Insert step $8'$ between steps 8 and 9 of Algorithm 3.2.2.

Step $8'$. If $n_1 = 1$ then go to Step 23. □

Similar reasoning leads us to modify Algorithm 3.2.4 to obtain Algorithm 3.2.6. The modifications (a), (b) and (c) to Algorithm 3.2.4 are the same as (a), (b) and (c) in Algorithm 3.2.5, but (d) is as follows.

(d) Insert Step $8'$ between steps 8 and 9 of Algorithm 3.2.4.

Step $8'$. If $n_1 = 1$ then go to Step 21 of Algorithm 3.2.4. □

Moreover, in the similar way Algorithm 3.2.7 will be obtained if we apply the modifications (a), (b) and (c) to Algorithm 3.2.2.

Under the conditions of Theorem 1.5.1, $\exists \hat{K}$ such that

$$\|g^{(K)}\| < \varepsilon_4 \qquad (\forall k \geqslant \hat{k}) ;$$

that is, from any starting point, the sequence $(x^{(K)})$ generated from algorithms 3.2.5 – 3.2.7 enters an $\varepsilon_4$ neighbourhood of a critical point, and ultimately one of the algorithms 3.2.2, 3.2.4 and 3.1.2 will be implemented.

3.3 Some Methods of Higher Order.

In view of the results reported by Wolfe (1978a) in connection with the iterative methods determined by (2.2.2), (2.2.3) and (2.2.6) - (2.2.7), M 2.2.1, M 2.2.2, it may be conjectured that iterative methods M 2.2.1 and M 2.2.2 and also, in the light of Theorem 2.2.3, M 2.2.3, may be used with advantage in place of the extended Newton method in modifying Algorithm 3.1.1. This can be done in a manner similar to that which has already been discussed in Section 3.2, to yield the following algorithms. We note that the following algorithms are supported by convergence theorems only if the Hessian of the objective functions are computed analytically.

Algorithm 3.3.1 (Corresponding to M 2.2.1).

Suppose that $x^{(0)}$, $p$, and $\varepsilon_i > 0$ $(i = 1, 2, 3)$ are given and $r = 2$.

Step 1. Set $k = 0$ and compute $F^{(k)} = F(x^{(k)})$.

Step 2. Compute $g^{(k)} = g(x^{(k)})$.

Step 3. Compute $G^{(k)} = G(x^{(k)})$.

Step 4. Form the Cholesky factorization of $G^{(k)}$ as in Algorithm 3.1.1 ($\bar{G}^{(k)} = L^{(k)} D^{(k)} L^{(k)T} = G^{(k)} + E^{(k)}$).

Step 5. If $\| g^{(k)} \|_2 > \varepsilon_1$, then go to Step 8.

Step 6. If $\|E^{(k)}\|_\infty = 0$, then $x^*$ is regarded as an adequate estimate of a strong local minimizer $x^*$ of F, and the algorithm is terminated.

Step 7. Determine $p^{(k)}$ by using (3.1.3) – (3.1.5), set $j = 0$, and go to Step 48.

Step 8. Determine $p^{(k)}$ from (3.1.2).

Step 9. If $\|p^{(k)}\|_2 \geq \varepsilon_3$, then go to Step 48.

Step 10. Set $j = 1$.

Step 11. Determine $\bar{x}^{(k)}$ and $\bar{F}^{(k)}$ from

$$\bar{x}^{(k)} = x^{(k)} + p^{(k)}$$

$$\bar{F}^{(k)} = F(\bar{x}^{(k)}).$$

Step 12. If $\bar{F}^{(k)} \leq F^{(k)} + \mu\, g^{(k)T} p^{(k)}$, where $\mu \in (0, 1/2]$, then go to Step 14.

Step 13. Go to Step 48.

Step 14. Compute $\bar{g}^{(k)} = g(\bar{x}^{(k)})$.

Step 15. If $\bar{g}^{(k)T} p^{(k)} < 0$ then go to Step 17.

Step 16. Set $j = 0$, $x^{(k+1)} = \bar{x}^{(k)}$, $g^{(k+1)} = \bar{g}^{(k)}$, $F^{(k+1)} = \bar{F}^{(k)}$, $k = k + 1$ and go to Step 3.

Step 17. Compute $\bar{p}^{(k)}$ from

$$L^{(k)} D^{(k)} L^{(k)T} \bar{p}^{(k)} = -\bar{g}^{(k)}.$$

Step 18. If $\|\bar{p}^{(K)}\|_2 > \varepsilon_3 r$ then go to Step 16.

Step 19. Set $\hat{x}^{(K)} = \bar{x}^{(K)} + \bar{P}^{(K)} / r$

Step 20. Compute $\hat{F}^{(K)} = F(\hat{x}^{(K)})$.

Step 21. If $\hat{F}^{(K)} \leqslant \bar{F}^{(K)} + \frac{\mu}{r} \bar{g}^{(K)T} p^{(K)}$ then go to Step 23.

Step 22. Set $p^{(K)} = \bar{p}^{(K)} + p^{(K)}$ and go to Step 48.

Step 23. If $|(F^{(K)} - \hat{F}^{(K)})/F^{(K)}| \geqslant 0.1$ then go to 25

Step 24. Set $x^{(K+1)} = \hat{x}^{(K)}$, $F^{(K+1)} = \hat{F}^{(K)}$, $j = 0$, $g^{(K+1)} = g(x^{(K+1)})$, $k = k + 1$ and go to Step 3.

Step 25. Compute $\hat{G}^{(K)} = G(\hat{x}^{(K)})$.

Step 26. Form the modified Cholesky factorization of $\hat{G}^{(K)}$,

$$\bar{\hat{G}}^{(K)} = \hat{L}^{(K)} \hat{D}^{(K)} \hat{L}^{(K)T}$$

Step 27. If $\min_{1 \leqslant i \leqslant n} \{ \hat{d}_i^{(K)} \} \leqslant \varepsilon_2$ then set $u = 2$ ;

otherwise set $u = 1$.

Step 28. Compute $\hat{p}^{(K)} = - \bar{\hat{G}}^{(K)-1} g^{(K)}$.

Step 29. If $\bar{g}^{(K)T} \hat{p}^{(K)} < 0$ then go to Step 35.

Step 30. If $j = 1$, then go to Step 32.

Step 31. Set $x^{(k+1)} = \bar{x}^{(k)}$, $g^{(k+1)} = \bar{g}^{(k)}$, $F^{(k+1)} = \bar{F}^{(k)}$, $j = 1$, $k = k + 1$, and go to Step 3.

Step 32. Set $x^{(k+1)} = \hat{x}^{(k)}$, $F^{(k+1)} = \hat{F}^{(k)}$, $\bar{G}^{(k+1)} = \hat{\bar{G}}^{(k)}$.

Step 33. Compute $g^{(k+1)} = g\,(x^{(k+1)})$.

Step 34. Set $k = k + 1$, and go to Step 5.

Step 35. Compute $\tilde{p}^{(k)} = -\hat{\bar{G}}^{(k)-1}\,\bar{g}^{(k)}$.

Step 36. If $\|\tilde{p}^{(k)}\|_2 \geqslant \varepsilon_3$ then go to Step 30.

Step 37. Set $\tilde{x}^{(k)} = \bar{x}^{(k)} + \tilde{p}^{(k)}$

Step 38. Compute $\tilde{F}^{(k)} = F\,(\tilde{x}^{(k)})$.

Step 39. If $\tilde{F}^{(k)} > \bar{F}^{(k)} + \mu\,\bar{g}^{(k)T}\,\hat{p}^{(k)}$ then go to Step 30.

Step 40. Set $j = j + 1$.

Step 41. If $u = 2$, or $j = P$ then go to Step 45.

Step 42. Set $\bar{x}^{(k)} = \tilde{x}^{(k)}$, $\bar{F}^{(k)} = \tilde{F}^{(k)}$.

Step 43. Compute $\bar{g}^{(k)} = g\,(\bar{x}^{(k)})$.

Step 44. Go to Step 29.

Step 45. Set $j = 1$, $x^{(K+1)} = \tilde{x}^{(K)}$, $F^{(K+1)} = \tilde{F}^{(K)}$, $k = k + 1$.

Step 46. Compute $g^{(K)} = g(x^{(K)})$.

Step 47. Go to Step 3.

Step 48. Determine $x^{(K+1)}$, $F^{(K+1)}$, and $g^{(K+1)}$ from

$$x^{(K+1)} = x^{(K)} + \alpha^{(K)} p^{(K)},$$

$$F^{(K+1)} = F(x^{(K+1)}),$$

$$g^{(K+1)} = g(x^{(K+1)}),$$

by using Algorithm 1.2.2.

Step 49. Set $k = k + 1$, and go to Step 3. □

Steps 12, and 13 ensure that if Newton's step with $\alpha^{(K)} = 1$ does not give a sufficient decrease, then Algorithm 1.2.2 is applied to find $\alpha^{(K)}$ and the modification in this iteration is not applied. The test in Step 15 ensures that the search direction $\bar{p}^{(K)}$ computed at Step 17 is down-hill at $x^{(K)}$, since

$$\bar{g}^{(K)T} p^{(K)} = -\bar{g}^{(K)T} \bar{G}^{(K)-1} g^{(K)}$$

$$= -\bar{p}^{(K)T} g^{(K)}. \qquad (3.3.1)$$

Steps 9, 18, and 36 contain a check on the step length, whereby the algorithm reverts to Newton's method with $\alpha^{(K)} = 1$ or method M 2.2.1

with extension $p_K < p$ if the length of $p^{(K)}$ is too large. One complete iteration corresponding to $p=3$ is shown in the Figure 3.1

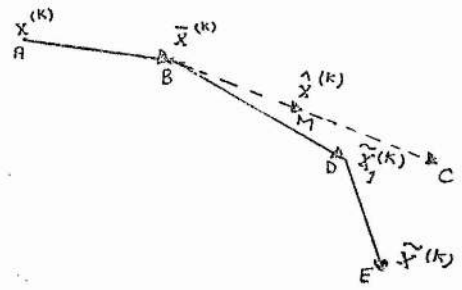



Figure 3.1

In Figure 3.1,

$$p^{(K)} = \overrightarrow{AB}$$

$$\bar{p}^{(K)} = \overrightarrow{BC}$$

$$\tilde{p}_1^{(K)} = \overrightarrow{BD}$$

$$\tilde{p}_2^{(K)} = \overrightarrow{DE}$$

$$\bar{x}^{(K)} = x^{(K)} + p^{(K)}, \quad \hat{x}^{(K)} = \bar{x}^{(K)} + \tfrac{1}{2}\bar{p}^{(K)},$$

$$\tilde{x}_1^{(K)} = \bar{x}^{(K)} + \tilde{p}_1^{(K)}, \text{ and } \tilde{x}_2^{(K)} = \tilde{X}_1^{(K)} + \tilde{p}_2^{(K)}.$$

In Figure 3.1, B, D, and E are the Newton iterate, the iterate corresponding to Method M 2.2.1 with $\gamma = 0$ and $\gamma = 1$ in (2.2.2), (2.2.3) respectively, while C represents the extended Newton iterate.

By steps 25, 26, 28, 29, and 35, we are sure that at each inner iteration $j$ (steps 29–44) $\tilde{p}^{(k)}$ is down-hill at $x^{(k)}$, since for all $j$ $(1 \leqslant j \leqslant p_k)$,

$$g^{(k)T} \tilde{p}^{(k)} = g^{(k)T} \left(- \bar{\hat{G}}^{(k)-1} \bar{g}^{(k)}\right)$$

$$= \bar{g}^{(k)T} \hat{\bar{p}}^{(k)} . \qquad (3.3.2)$$

Also, if an iteration ends at B or M or C, we still have

$$g^{(k)T} \tilde{p}^{(k)} < 0, \qquad (3.3.3)$$

and

$$g^{(k)T} \bar{p}^{(k)} < 0, \qquad (3.3.4)$$

and the corresponding sufficient decrease condition is satisfied. Therefore, analogues of Theorem 3.2.1 and Theorem 3.2.2 may be proved.

By Algorithm 3.3.1, one of the requirements for the extension from D to E is to evaluate the function at one extra point, say E, to test whether

$$F(\tilde{x}_2^{(k)}) \leq F(\tilde{x}_1^{(k)}) + \mu\, g(x^{(k)})^T \tilde{p}_2^{(k)} \qquad (3.3.5)$$

is satisfied. It is reasonable to determine the total step $p^{(k)}$ without requiring (3.3.5). That is, for $P = 3$, Algorithm 1.2.2 is applied to find $\alpha^{(k)}$ along $\overrightarrow{AE} = p^{(k)} + \tilde{p}_1^{(k)} + \tilde{p}_2^{(k)}$. This gives rise to the following algorithm.

Algorithm 3.3.2 (corresponding to M 2.2.1)

Steps 1 - 28 are the same as in Algorithm 3.3.1.

Step 29. Set $s^{(k)} = p^{(k)}$ and $j = 1$.

Step 30 If $\bar{g}^{(k)T} \hat{p}^{(k)} < 0$ then go to Step 35.

Step 31. If $j \neq 1$ then go to Step 48 of Algorithm 3.3.1.

Step 32. Set $x^{(k+1)} = \hat{x}^{(k)}$, $F^{(k+1)} = \hat{F}^{(k)}$.

Step 33. Compute $g^{(k+1)} = g(x^{(k+1)})$.

Step 34. Set $k = k + 1$, $\bar{G}^{(k+1)} = \hat{\bar{G}}^{(k)}$ and go to Step 5.

Step 35. Compute $\tilde{p}^{(k)} = -\hat{\bar{G}}^{(k)-1} \bar{g}^{(k)}$.

Step 36. Set $s^{(k)} = s^{(k)} + \tilde{p}^{(k)}$.

Step 37. If $\| s^{(k)} \|_2 < \varepsilon_3$ then go to Step 39.

Step 38. Set $p^{(k)} = s^{(k)}$ and go to Step 48 of Algorithm 3.3.1.

Step 39. Set $p^{(k)} = s^{(k)}$ and $j = j + 1$.

Step 40. If $j = p$ or $u = 2$ then go to Step 48 of Algorithm 3.3.1.

Step 41. Set $\bar{x}^{(k)} = x^{(k)} + p^{(k)}$.

Step 42. Compute $\bar{g}^{(K)} = g(\bar{x}^{(K)})$ and go to Step 30. □

In Algorithm 3.3.1 and therefore in Algorithm 3.3.2, if we set $\mathfrak{F} = 1$, then the algorithms 3.3.3 and 3.3.4 corresponding to Method M 2.2.3 will be obtained.

The minimization algorithm corresponding to M 2.2.2 is as follows.

Algorithm 3.3.5 (corresponding to M 2.2.2)

Suppose that $x^{(0)}$, p, $\varepsilon_i > 0$ $(i = 1, 2, 3)$ are given.

Step 1. Set $k = 0$ and compute $F^{(K)} = F(x^{(K)})$.

Step 2. Compute $g^{(K)} = g(x^{(K)})$.

Step 3. Compute $G^{(K)} = G(x^{(K)})$.

Step 4. Determine the modified Cholesky factorization of $G^{(K)}$ as in Algorithm 3.1.1.

Step 5. If $\|g^{(K)}\|_2 > \varepsilon_1$, then go to Step 8.

Step 6. If $\|E^{(K)}\|_\infty = 0$, then $x^{(K)}$ is regarded as an adequate estimate of a strong local minimizer $x^*$ of F, and the algorithm is terminated.

Step 7. Determine $p^{(K)}$ by using (3.1.3) - (3.1.5), and go to Step 49.

Step 8. Determine $p^{(k)}$ from (3.1.2).

Step 9. If $\|p^{(k)}\|_2 \geqslant \varepsilon_3$ then go to Step 49.

Step 10. Set $\hat{x}^{(k)} = x^{(k)} + p^{(k)}$.

Step 11. Compute $\hat{F}^{(k)} = F(\hat{x}^{(k)})$.

Step 12. If $\hat{F}^{(k)} > F^{(k)} + \mu\, g^{(k)T} p^{(k)}$, where $\mu \in (0, \frac{1}{2}]$, then go to Step 49.

Step 13. If $\left|(F^{(k)} - \hat{F}^{(k)}) / F^{(k)}\right| > 0.1$ then go to Step 17.

Step 14. Set $x^{(k+1)} = \hat{x}^{(k)}$, $F^{(k+1)} = \hat{F}^{(k)}$.

Step 15. Compute $g^{(k+1)} = g(x^{(k+1)})$.

Step 16. Set $k = k + 1$, and go to Step 3.

Step 17. Compute $\hat{G}^{(k)} = G(\hat{x}^{(k)})$.

Step 18. Set $\hat{\hat{G}}^{(k)} = G^{(k)} + \hat{G}^{(k)}$.

Step 19. Determine the modified Cholesky factorization of $\hat{\hat{G}}^{(k)}$ as in Algorithm 3.1.1.

Step 20. Compute $\bar{p}^{(k)} = -\hat{\hat{G}}^{(k)-1} g^{(k)}$.

Step 21. If $\|\bar{p}^{(K)}\| < \varepsilon_3/2$ then go to Step 25.

Step 22. Set $x^{(K+1)} = \hat{x}^{(K)}$, $F^{(K+1)} = \hat{F}^{(K)}$.

Step 23. Compute $g^{(K+1)} = g(x^{(K+1)})$.

Step 24. Set $k = k + 1$, $G^{(K)} = \hat{G}^{(K)}$ and go to Step 4.

Step 25. Determine the modified Cholesky factorization of $\hat{G}^{(K)}$ as in Algorithm 3.1.1. That is

$$\hat{\bar{G}}^{(K)} = \hat{L}^{(K)} \hat{D}^{(K)} \hat{L}^{(K)T}$$

$$= \hat{G}^{(K)} + E^{(K)}.$$

Step 26. If $\min_{1 \leq i \leq n} \{ \hat{d}_i^{(K)} \} \leq \varepsilon_2$, then set $u = 2$;
Otherwise set $u = 1$.

Step 27. Set $\bar{x}^{(K)} = x^{(K)} + 2\bar{p}^{(K)}$.

Step 28. Compute $\bar{F}^{(K)} = F(\bar{x}^{(K)})$.

Step 29. If $\bar{F}^{(K)} < \hat{F}^{(K)}$ and $\bar{F}^{(K)} \leq F^{(K)} + 2\mu g^{(K)T} \bar{p}^{(K)}$ then go to Step 33.

Step 30. Set $x^{(K+1)} = \hat{x}^{(K)}$, $F^{(K+1)} = \hat{F}^{(K)}$, $\bar{G}^{(K)} = \hat{\bar{G}}^{(K)}$.

Step 31. Compute $g^{(K+1)} = g(x^{(K+1)})$.

Step 32. Set $k = k + 1$ and go to Step 5.

Step 33. Compute $\hat{p}^{(k)} = -\hat{G}^{(k)-1} g^{(k)}$.

Step 34. Compute $\bar{g}^{(k)} = g(\bar{x}^{(k)})$ and set $j = 1$.

Step 35. If $\bar{g}^{(k)T} \hat{p}^{(k)} < 0$ then go to Step 38.

Step 36. If $j = 1$ then go to Step 30.

Step 37. Set $x^{(k+1)} = \bar{x}^{(k)}$, $g^{(k+1)} = \bar{g}^{(k)}$, $F^{(k+1)} = \bar{F}^{(k)}$, $k = k + 1$, and go to Step 3.

Step 38. Compute $\tilde{p}^{(k)} = -\hat{G}^{(k)-1} \bar{g}^{(k)}$.

Step 39. If $\| \tilde{p}^{(k)} \|_2 \geq \varepsilon_3$ then go to Step 36.

Step 40. Set $\tilde{x}^{(k)} = \bar{x}^{(k)} + \tilde{p}^{(k)}$.

Step 41. Compute $\tilde{F}^{(k)} = F(\tilde{x}^{(k)})$.

Step 42. If $\tilde{F}^{(k)} > \bar{F}^{(k)} + \mu \bar{g}^{(k)T} p^{(k)}$ then go to Step 36.

Step 43. Set $j = j + 1$.

Step 44. If $j = p$ or $u = 2$ then go to Step 47.

Step 45. Set $\bar{x}^{(k)} = \tilde{x}^{(k)}$, $\bar{F}^{(k)} = \tilde{F}^{(k)}$.

Step 46. Compute $\bar{g}^{(k)} = g(\bar{x}^{(k)})$ and go to Step 35.

Step 47. Set $x^{(k+1)} = \tilde{x}^{(k)}$, $F^{(k+1)} = \tilde{F}^{(k)}$.

Step 48. Compute $g^{(k+1)} = g(x^{(k+1)})$ and go to Step 3.

Step 49. Determine $x^{(k+1)}$, $F^{(k+1)}$, and $g^{(k+1)}$ from

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)},$$

$$F^{(k+1)} = F(x^{(k+1)}),$$

$$g^{(k+1)} = g(x^{(k+1)}),$$

by using Algorithm 1.2.2.

Step 50. Set $k = k + 1$ and go to Step 3. □

The geometrical representation of a typical iteration corresponding to $p = 3$ is shown in Figure 3.2.

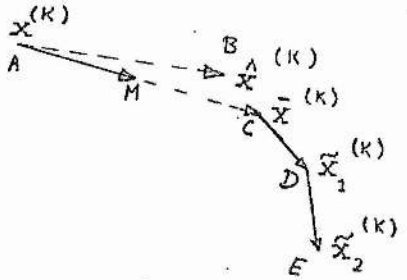



Figure 3.2

In Figure 3.2

$$p^{(k)} = \overrightarrow{AB}$$

$$= -G(x^{(k)})^{-1} g(x^{(k)}),$$

$$\bar{p}^{(K)} = \overrightarrow{AM}$$

$$= -\left(G\left(x^{(K)}\right) + G\left(\hat{x}^{(K)}\right)\right)^{-1} g\left(x^{(K)}\right),$$

$$\tilde{p}_1^{(K)} = \overrightarrow{CD}$$

$$= -G\left(\hat{x}^{(K)}\right)^{-1} g\left(\bar{x}^{(K)}\right)$$

$$\tilde{p}_2^{(K)} = -G\left(\hat{x}^{(K)}\right)^{-1} g\left(\tilde{x}_1^{(K)}\right)$$

$$= \overrightarrow{DE}$$

$$\hat{x}^{(K)} = x^{(K)} + p^{(K)}, \quad \bar{x}^{(K)} = x^{(K)} + 2\bar{p}^{(K)}, \quad \tilde{x}_1^{(K)} = \bar{x}^{(K)} + \tilde{p}_1^{(K)},$$

$$\tilde{x}_2^{(K)} = \tilde{x}_1^{(K)} + \tilde{p}_1^{(K)}, \text{ and } x^{(K+1)} = \tilde{x}_2^{(K)}.$$

From Algorithm 3.3.5 it is easily verified that at each iteration, sufficient decrease of the objective function has been made, and also, all search directions involved at each iteration are down-hill at $x^{(K)}$. Thus, analogues of theorems 3.2.1 and 3.2.2 may be proved for the sequence $(x^{(K)})$ generated from this algorithm.

The method corresponding to M 2.2.2 may be implemented in two different ways as for M 2.2.1 , without altering the convergence properties. The second implementation corresponding to M 2.2.2 is as follows.

Algorithm 3.3.6 (Corresponding to M 2.2.2)

Steps 1 - 34 are the same as in Algorithm 3.3.5.

Step 35. Set $s^{(K)} = 2\bar{p}^{(K)}$.

Step 36. If $\bar{g}^{(K)T} \hat{p}^{(K)} < 0$ then go to Step 39.

Step 37. If $j = 1$ then go to Step 30.

Step 38. Go to Step 49 of Algorithm 3.3.5

Step 39. Compute $\tilde{p}^{(K)} = -\hat{G}^{(K)-1}\,\bar{g}^{(K)}$.

Step 40. Set $s^{(K)} = s^{(K)} + \tilde{p}^{(K)}$.

Step 41. If $\|s^{(K)}\| \leq \varepsilon_3$ then go to Step 43.

Step 42. Set $p^{(K)} = s^{(K)}$ and go to Step 49 of Algorithm 3.3.5.

Step 43. Set $p^{(K)} = s^{(K)}$.

Step 44. Set $j = j + 1$.

Step 45. If $j = P$ or $u = 2$, then go to Step 49 of Algorithm 3.3.5

Step 46. Set $\bar{x}^{(K)} = x^{(K)} + p^{(K)}$.

Step 47. Compute $\bar{g}^{(K)} = g(\bar{x}^{(K)})$ and go to Step 36. □

# Chapter 4.

## On the Numerical Estimation of Hessian Matrices.

### 4.1 Introduction.

In order to implement Algorithm 3.1.2 when analytical formulae for the second partial derivatives are not available, Gill, Murray, and Picken (1974) have used (3.1.6) and (3.1.7), in which the analytical formulae for the first partial derivatives of the objective function are known. The method requires n evaluations of the gradient vector $g$ of the objective function $F : R^n \longrightarrow R^1$ at $x \in R^n$. Computational experience shows that usually, the results which are obtained by using Algorithm 3.1.1 are identical, to a high degree of precision, to those which are obtained by using Algorithm 3.1.2.

Gill and Murray (1974c) have suggested that algorithms 3.1.1 and 3.1.2 may be used for the solution of large-scale unconstrained optimization problems if the Hessian matrix is sparse. In particular, they propose a method for estimating $G(x)$ numerically when $G$ is m-diagonal. This method requires m evaluations of $g$ to estimate $G(x)$ if it is supposed that $g(x)$ is already known. Computational experience shows that usually the results which are obtained by using this method with Algorithm 3.1.2 to minimize objective functions with m - diagonal Hessian matrices are identical, to a high degree of precision, to the results which are obtained by using Algorithm 3.1.1, in which G is computed analytically.

In view of the preceding remarks, it is desirable to consider whether more efficient methods for the numerical estimation of both full and m - diagonal Hessian matrices may be devised. Clearly, to be satisfactory, any such methods should be as accurate and as reliable

as those which have been proposed in the previously-mentioned references.

In this chapter, a number of algorithms for the numerical estimation of the Hessian matrix G of an objective function $F : R^n \longrightarrow R^1$ are described. In each case it is supposed that analytical formulae for F and for the n components of the gradient g of F are known. It is often true that the formulae for the components of g contain expressions in common, so that the computational labour which is required in order to evaluate g is often less than n times that which is required for the evaluation of one component of g. Furthermore, it is often true that the formulae for F contains expressions which are common to some or all of the formulae for the components of g. In this case a considerable saving in computational labour can be made in the simultaneous evaluation of F and g.

## 4.2 The Estimation of Full Hessians

In this section a very simple algorithm for estimating full Hessian matrices is described. It would appear to have a slight advantage over the method of Gill, Murray, and Picken (1974) as regards computing time, while giving virtually identical numerical results when used with Algorithm 3.1.2.

The n × 1 gradient vector g (x) and the n × n Hessian matrix G (x) of $F : R^n \longrightarrow R^1$ at x are defined by

$$g_i(x) = \partial_i F(x) \qquad (i = 1, 2, \ldots, n),$$

and

$$G_{ij}(x) = \partial_j \partial_i F(x) \qquad (i, j, = 1, 2, \ldots, n),$$

where

$$\partial_i F(x) = \frac{\partial}{\partial x_i} F(x) \qquad (i = 1, 2, \ldots, n)$$

and $$\partial_j \partial_i F(x) = \frac{\partial^2}{\partial x_j \partial x_i} F(x) \quad (i, j = 1, 2, \ldots, n).$$

Let $h > 0$ be a given step length. Then the $G(x)$ may be estimated with truncation error $O(h)$ according to

$$G_{ij} \approx \frac{1}{h} (g_i(x + h e_j) - g_i(x)) \quad (i, j = 1, 2, \ldots, n), \tag{4.2.1}$$

where, for $j = 1, 2, \ldots, n$, $e_j$ is column $j$ of the $n \times n$ unit matrix.

Gill and Murray argue that because of truncation and rounding errors, the estimate of $G(x)$ which is obtained by using (4.2.1) as it stands may not be symmetric. They therefore advocate symmetrizing the estimate of $G(x)$ by using (3.1.7). Gill and Murray have discovered, somewhat surprisingly, that the value of $h$ which should be used in (4.2.1) is rather insensitive to $F$ but depends upon the computing machine precision. They suggest that a suitable value for $h$ should be between $2^{-\frac{3t}{2}}$ and $2^{-\frac{t}{2}}$, where $t$ is the number of bits in the word length of the computing machine. Their algorithm is as follows.

Algorithm 4.2.1

Let $x$, $g = g(x)$, $n$, and $h$ be given.

Step 1. Set $j = 1$.

Step 2. Compute $g^+ = g(x + he_j)$.

Step 3. Compute $G_{ij} = (g_i^+ - g_i) / h \quad (i = 1, 2, \ldots, n)$.

Step 4. If $j = n$, then go to Step 6.

Step 5. Set $j = j + 1$ and go to Step 2.

Step 6. Set $G_{ij} = (G_{ij} + G_{ji}) / 2$ $(j = 1, 2, \ldots, n - 1)$; $i = j + 1, \ldots, n)$.

Step 7. Stop. □

Algorithm 4.2.1 leaves a symmetrized estimate of G (x) in the lower triangle of G. Algorithm 4.2.1 requires n evaluations of g if it is supposed that g (x) is known.

The following algorithm requires n calls of the subprogram for g as does Algorithm 4.2.1, but fewer arithmetical operations are required.

Algorithm 4.2.2

Let x, $g = g(x)$, n, and h be given.

Step 1. Set $j = 1$.

Step 2. Compute $g_{ij} = g_i(x + he_j)$ $(i = j, \ldots, n)$.

Step 3. Compute $G_{ij} = (g_{ij} - g_i) / h$ $(i = j, \ldots, n)$.

Step 4. If $j = n$, then go to Step 6.

Step 5. Set $j = j + 1$ and go to Step 2.

Step 6. Stop. □

Algorithm 4.2.2 leaves an estimate of G (x) in the lower triangle of G.

## 4.3 The Estimation of m - diagonal Hessians

If the n × n Hessian matrix G (x) is m - diagonal so that $G_{ij}(x) = 0$ ( $|i - j| > (m - 1) / 2$), then Gill and Murray (1974c) suggest an algorithm for the estimate of G (x), in which only m evaluations of g are required if it is supposed that g (x) is already known. The algorithm is based upon the fact that if $y_k$ (k = 1, 2, ..., m) are defined by

$$y_k = \left\{ g\left(x + h \sum_{l=0}^{l_k} e_{k+lm}\right) - g(x) \right\} / h, \qquad (4.3.1)$$

where

$$l_k = \left[ (n - k) / m \right], \qquad (4.3.2)$$

in which [u] is the greatest integer not greater than u, then for i = 1, 2, ..., n and k = 1, 2, ..., m,

$$y_{ik} = \sum_{l=0}^{l_k} G_{i,k+lm} + O(h), \qquad (4.3.3)$$

where

$$y_k = (y_{1k}, \ldots, y_{nk})^T.$$

From (4.3.3) it follows that for j = 1, 2, ..., n,

$$G_{ij} = y_{ik} + O(h), \qquad (4.3.4)$$

where

$$k = j - [(j - 1)/m]m, \tag{4.3.5}$$

and

$$i = \begin{cases} 1, 2, \ldots, \{(m + 1)/2 + j - 1\} & (1 \leqslant j \leqslant (m + 1)/2) \\ \{j - (m - 1)/2\}, \ldots, \{j + (m - 1)/2\} & ((m + 3)/2 \leqslant j \leqslant n - (m-1)/2) \\ \{j - (m - 1)/2\}, \ldots, n & (n - (m - 3)/2 \leqslant j \leqslant n). \end{cases} \tag{4.3.6}$$

If, in (4.3.4), the O (h) term is neglected, then the following algorithm is obtained.

Algorithm 4.3.1 (a)

Let x, m, n, h, and g (x) be given.

Step 1. For $k = 1, \ldots, m$, compute $y_k$ from (4.3.1) and (4.3.2).

Step 2. Set $j = 1$.

Step 3. Compute k from (4.3.5).

Step 4. If $j \leqslant (m + 1)/2$ then go to Step 7.

Step 5. If $j \leqslant n - (m - 1)/2$ then go to Step 8.

Step 6. For $i = \{j - (m - 1)/2\}, \ldots, n$, set $G_{ij} = y_{iK}$, and go to Step 9.

Step 7. For $i = 1, 2, \ldots, (j + (m - 1)/2)$, set $G_{ij} = y_{iK}$ and go to Step 9.

Step 8. For $i = (j - (m - 1)/2), \ldots, (j + (m - 1)/2)$, set $G_{ij} = y_{iK}$

Step 9. If $j = n$, then go to Step 11.

Step 10. Set $j = j + 1$ and go to Step 3.

Step 11. Stop. □

If it is required to obtain a symmetrized estimate of G (x), then Step 11 of Algorithm 4.3.1 (a) must be replaced with the following, to give Algorithm 4.3.1 (b).

Step 11. Set $i = 2$.

Step 12. If $i \leq (m + 1)/2$, then go to Step 14.

Step 13. For $j = (i - (m + 1)/2), \ldots, (i - 1)$, set $G_{ij} = (G_{ij} + G_{ji})/2$, and to to Step 15.

Step 14. For $j = 1, 2, \ldots, (i - 1)$, set $G_{ij} = (G_{ij} + G_{ji})/2$.

Step 15. If $i = n - 1$, then go to Step 17.

Step 16. Set $i = i + 1$ and go to Step 12.

Step 17. Stop. □

Steps 11 - 17 leave a symmetrized estimate of G (x) in the lower triangle of G.

Let $r = (m + 1)/2$, and for $j = 1, 2, \ldots, r$

let $y_j$ be defined by

$$y_j = \frac{1}{h} \left( g \left( x + \sum_{l=0}^{l_j} h\, e_{j+lr} \right) - g(x) \right) \qquad (4.3.7),$$

where

$$l_j = \left[ (n - j)/r \right]. \qquad (4.3.8)$$

Then for $i = 1, 2, \ldots, n$, and $j = 1, 2, \ldots, r$,

$$y_{ij} = \sum_{l=0}^{l_j} G_{i, lr+j} + O(h) . \qquad (4.3.9)$$

If the term O (h) is neglected, then the following results may be deduced from (4.3.9).

For $i = 1, 2, \ldots, n$,

$$G_{ii} = y_{is} , \qquad (4.3.10)$$

where $s = i - \left[ (i - 1)/r \right] r.$ (4.3.11)

For $j = i + 1, \ldots, i + r - 1$,

$$G_{ij} = \begin{cases} y_{it} & (j - r < 1) \\ y_{it} - G_{i,\,j-r} & (j - r \geqslant 1) \end{cases} \tag{4.3.12}$$

where $t = j - \left[ (j - 1)/r \right] r.$ (4.3.13)

If it is assumed that $G_{ji} = G_{ij}$, then the following algorithm is obtained.

Algorithm 4.3.2 (a)

Let x, m, n, h, and g (x) be given.

Step 1. Set $r = (m + 1)/2$.

Step 2. Compute $y_j$ $(j = 1, 2, \ldots, r)$ from (4.3.7), and (4.3.8).

Step 3. Compute $G_{ii}$ $(i = 1, 2, \ldots, n)$ from (4.3.10), and (4.3.11).

Step 4. Set $i = 1$.

Step 5. Set $j = i + 1$.

Step 6. Compute t from (4.3.13).

Step 7. If $j - r < 1$, then set $G_{ij} = y_{it}$, and go to Step 9.

Step 8. Set $G_{ij} = y_{it} - G_{i,j-r}$

Step 9. Set $G_{ji} = G_{ij}$

Step 10. If $j = i + r - 1$, then go to Step 12.

Step 11. Set $j = j + 1$ and go to Step 6.

Step 12. If $i = n - 1$, then go to Step 14.

Step 13. Set $i = i + 1$ and go to Step 5.

Step 14. Stop. □

The estimate of G (x) which is obtained by using Algorithm 4.3.2 (a) is forced to be symmetric by Step 9. Another estimate of G (x) could be obtained by using Algorithm 4.3.2 (a) "in reverse," and the two estimates could then be "averaged" as in Step 14 of Algorithm 4.3.1 (b). The suggestion is that Step 14 of Algorithm 4.3.2 (a) should be replaced with the following, to give Algorithm 4.3.2 (b).

Step 14. Set $i = n$.

Step 15. Set $j = i - r + 1$.

Step 16. Compute t from (4.3.13).

Step 17. If $j + r > n$, then set $\bar{G}_{ij} = y_{it}$, and go to Step 19.

Step 18. Set $\bar{G}_{ij} = y_{it} - \bar{G}_{i,j+r}$.

Step 19. Set $\bar{G}_{ji} = \bar{G}_{ij}$.

Step 20. If $j = i - 1$, then go to Step 22.

Step 21. Set $j = j + 1$ and go to Step 16.

Step 22. If $i = 2$, then go to Step 24.

Step 23. Set $i = i - 1$ and go to Step 15.

Step 24. For $i = 1, 2, \ldots, n - 1$, and $j = i + 1, \ldots, \min\{i + r, n + 1\}$, set $G_{ij} = (G_{ij} + \bar{G}_{ij})/2$.

Step 25. Stop. □

Algorithms 4.3.2 (a) and 4.3.2 (b) each require $(m + 1)/2$ evaluations of g.

The ideas underlying Algorithm 4.3.2 (b) may be used to obtain the following algorithm, which also requires $(m + 1)/2$ evaluations of g.

Algorithm 4.3.3 (b).

Let x, m, n, h, and g (x) be given.

Step 1. Set $r = (m + 1)/2$

Step 2. Compute $y_j$ $(j = 1, 2, \ldots, r)$ from (4.3.7), and (4.3.8).

Step 3. Compute $G_{ii}$ $(i = 1, 2, \ldots, n)$ from (4.3.10), and (4.3.11).

Step 4. Set $i = 1$.

Step 5. Compute $k = \min \{ i + r, n + 1 \}$ .

Step 6. Set $j = i + 1$.

Step 7. Compute $l_1 = [(i - 1)/r]$.

Step 8. Compute $l_2 = \min \{ [i/r], [(j - r - 1)/r] \}$ .

Step 9. Compute $l_3 = [(n - j)/r]$

Step 10. Compute $l_4 = \min \{ l_3 , [(n - r - i)/r] \}$.

Step 11. Compute $s$ from (4.3.11).

Step 12. Compute $t$ from (4.3.13).

Step 13. Compute $S_1 = \sum_{l=0}^{l_1} y_{i-rl, t}$

Step 14. If $l_2 < 0$, then set $S_2 = 0$ and go to Step 16.

Step 15. Compute $S_2$ from

$$S_2 = \sum_{l=0}^{l_2} y_{i-rl,\,s}$$

Step 16. Compute $S_3$ from

$$S_3 = \sum_{l=0}^{l_3} y_{i+rl,\,s}$$

Step 17. If $l_4 < 0$, then set $s_4 = 0$ and go to Step 19.

Step 18. Compute $S_4$ from

$$S_4 = \sum_{l=0}^{l_4} y_{i+r(l+1),\,t}$$

Step 19. Compute $G_{ij}$ from

$$G_{ij} = (S_1 + S_3 - S_2 - S_4)/2.$$

Step 20. Set $G_{ji} = G_{ij}$.

Step 21. If $j = k - 1$, then go to Step 23.

Step 22. Set $j = j + 1$, and go to Step 6.

Step 23. If $i = n - 1$, then go to Step 25.

Step 24. Set $i = i + 1$ and go to Step 5.

Step 25. Stop. □

This algorithm leaves a symmetrized estimate of G (x) in the principal diagonal and in the adjacent (r - 1) diagonals of G.

If steps 9 and 10 of Algorithm 4.3.3 (b) are deleted, and if steps 16 - 19 of Algorithm 4.3.3 (b) are deleted and are replaced with

Step 16. Compute $G_{ij}$ from

$$G_{ij} = (S_1 - S_2)/2 ,$$

then Algorithm 4.3.3 (a), corresponding to Algorithm 4.3.2 (a) is obtained.

The preceding algorithm is based upon a theorem which is proved subsequently. We require the following lemmas. We note that G (x) is an m - diagonal matrix and that $r = (m + 1)/2$.

Lemma 4.3.1

If $t = i - [(i - 1)/r]\, r \quad (1 \leqslant i \leqslant n)$ then

$$G_{ii} = y_{it} .$$

Proof

By the choice of t, we have

$i = rl' + t$ where $1 \leqslant t \leqslant r$, and

$l' = [(i-1)/r]$.

Let $j_1 = rl'' + t$, where $l''$ is any positive integer. Then, when $l'' \neq l'$, we have

$$|i - j_1| = |l' - l''| r \geqslant r .$$

Thus, by (4.3.9) we have $G_{i,i} = y_{i,t}$ . □

Lemma 4.3.2

Let

$$j < i < j + r ,$$

and

$$t = j - \left[ (j - 1)/r \right] r \quad (1 \leqslant j \leqslant n).$$

1. If $j + r \leqslant n$ then

$$G_{i,j} + G_{i,j+r} = y_{i,t} .$$

2. If $j + r > n$ then

$$G_{i,j} = y_{i,t} .$$

Proof

1. By the definition of t, we have

$$j = rl' + t, \text{ where } 1 \leqslant t \leqslant r, \text{ and}$$
$$l' = [(j-1)/r].$$

Let

$j_1 = rl + t$, where $l$ is any integer $> 0$. Since $j < i < j + r$,

then $|i - j_1| \geqslant r \quad (\forall l$ such that $|l - l'| > 1$ or $l = l' - 1)$.

Therefore $G_{i,j_1} = 0 \quad (\forall l$ such that $|l - l'| > 1$ or $l = l' - 1)$.

For $l = l'$ or $l = l' + 1$ we have $j_1 = j$ or $j_1 = j + r$ respectively. Therefore by (4.3.9), we have

$$G_{i,j} + G_{i,j+r} = y_{i,t}.$$

2. The proof of Statement 2 is similar to that of Statement 1 and will therefore be omitted. □

Lemma 4.3.3

If $0 < j - r < i < j \leqslant n \quad (j = r, r + 1, \ldots, n)$

then

$$G_{i,j} = G_{i-r,j-r} + (y_{i,t_j} - y_{j-r,t_i}), \tag{4.3.14}$$

where

$$t_s = s - \left[(s - 1)/r\right] r. \quad (s = 1, 2, \ldots,) \tag{4.3.15}$$

Proof.

Since $0 < j - r < i < j \leqslant n$ then by Lemma 4.3.2 and (4.3.15) we have

$$G_{i,j-r} + G_{i,j} = y_{i,t_j}, \tag{4.3.16}$$

and

$$G_{j-r,i} + G_{j-r,i-r} = y_{j-r,t_i}. \tag{4.3.17}$$

Since G is symmetric then from (4.3.16) and (4.3.17) Equation (4.3.14) holds. □

Theorem 4.3.1

If 1. $G_{ij} = 0$ for all i, j such that $i \leqslant 0$ or $j \leqslant 0$ or $n < i$ or $n < j$.

2. $y_{ij} = 0$ for all i, j such that $i \leqslant 0$ or $j \leqslant 0$.

3. $1 \leqslant i < j \leqslant n$ and $j < i + r$ $(1 \leqslant j \leqslant n)$,

then

$$G_{ij} = \sum_{l=0}^{l_1} \left\{ y_{i-rl,\, t_j} - y_{j-r(l+1),\, t_i} \right\} \qquad (4.3.18)$$

where

$t_s$ $(1 \leqslant s)$ is defined by (4.3.15),

and $l_1 = [i/r]$.

Proof.

We proceed by induction on i.

Let $i = 1$. Then $l_1 = [i/r] = 0$, $t_i = 1$ and then by Hypothesis 1, $y_{j-r,\, i} = 0$ because $j - r \leqslant i - 1 = 0$. Also by (4.3.9)

we have

$$y_{1j} = G_{1j} + G_{1,j+r} + \ldots + 0\,(h)$$

$$= G_{1,j} + 0\,(h).$$

That is

$$G_{1,j} = y_{1,t_j} - y_{j-r,\, t_i},$$

where

$$t_j = j.$$

Therefore (4.3.18) holds for $i = 1$.

Assume that (4.3.18) holds for all integer $< i$. Then by

Lemma 4.3.3 we have

$$G_{i,j} = G_{i-r,j-r} + (y_{i,t_j} - y_{j-r,t_i}). \qquad (4.3.19)$$

Since $i - r < i$ and $i - r < j - r$, then by (4.3.18) we have

$$G_{i-r,j-r} = \sum_{1=0}^{1_1} \left\{ y_{i-r-r1,t'} - y_{j-r(1+2),t''} \right\},$$

where $1_1 = [(i - r)/r]$, $t' = t_{j-r}$ and $t'' = t_{i-r}$

But since $t_s = t_{s-r}$ $(1 \leqslant s)$ and $[(i - r)/r] = [i/r] - 1$, then

$$G_{i-r,j-r} = \sum_{1=0}^{1_2-1} \left\{ y_{i-r(1+1),t_j} - y_{j-r(1+2),t_i} \right\} \qquad (4.3.20)$$

where $1_2 = [i/r]$.

Thus, by (4.3.19), (4.3.20)

we have

$$G_{i,j} = \sum_{1=0}^{1_2} \left\{ y_{i-r1,t_j} - y_{j-r(1+1),t_i} \right\}. \quad \square$$

Lemma 4.3.4

If $1 \leqslant j < i < j + r \leqslant n$ then

$$G_{i,j} = G_{i+r,j+r} + (y_{i,t_j} - y_{j+r,t_i}),$$

where $t_s (1 \leqslant s)$ is defined by (4.3.15).

Proof.

The proof is similar to that of Lemma 4.3.3, and is therefore omitted. □

Theorem 4.3.2

If 1. hypotheses 1 and 2 of Theorem 4.3.1 hold;

2. $1 \leqslant j < i \leqslant n$ and $i < j + r$,

then

$$G_{i,j} = \sum_{l=0}^{l_1} \left\{ y_{i+rl,\, t_j} - y_{j+r(l+1),\, t_i} \right\},$$

where $t_s$ $(1 \leqslant s)$ is defined as in (4.3.15) and

$$l_1 = \lfloor (n - i)/r \rfloor.$$

Proof.

Set $k = n - i$ $i = n, (n - 1), \ldots, 1$. The proof is by induction on k, and it is similar to that of Theorem 4.3.1, if we apply Lemma 4.3.4; thus the proof is omitted. □

In Algorithm 4.3.3 the diagonal elements of G are determined by using Lemma 4.3.1, while theorems 4.3.1 and 4.3.2 are used to provide the upper triangular and lower triangular part of G respectively.

The following theorem guarantees that the elements $y_{s,t}$ $(1 \leqslant s \leqslant n,\ 1 \leqslant t \leqslant r)$ which are applied to compute $G_{i,j}$ are different from those which are used to compute $G_{j,i}$ $(\forall i, j \leqslant n,\ i \neq j)$.

Theorem 4.3.3

If $1 \leqslant i < j < i + r \leqslant n$ then the two expressions for approximating $G_{i,j}$ and $G_{j,i}$ have no common elements.

Proof.

Since $i < j$, then by Theorem 4.3.1 we have

$$G_{i,j} = \sum_{l=0}^{l_1} \left\{ y_{i-rl,\, t_j} - y_{j-r(l+1),\, t_i} \right\} , \qquad (4.3.21)$$

where $t_s$ $(1 \leqslant s)$ is defined as (4.3.15) and $l_1 = [i/r]$.

Now set $i' = j$ and $j' = i$. Then $t_{i'} = t_j$ and $t_{j'} = t_i$. Moreover, since $j' < i'$, and $i' < j' + r$, then by Theorem 4.3.2 we have

$$G_{j,i} = \sum_{l=0}^{l_2} \left\{ y_{j+rl,\, t_i} - y_{i+r(l+1),\, t_j} \right\} , \qquad (4.3.22)$$

where $l_2 = [(n - j)/r]$.

Suppose that at least one of the terms in (4.3.21) appears in (4.3.22). Since $i < j + r$ and therefore $t_i \neq t_j$, then there exist $l, l' \geqslant 0$ such that either

1. $i - rl = i + r(l' + 1)$,

or

2. $j - r(l + 1) = j + rl'$.

Obviously both cases 1 and 2 are impossible. □

By theorems 4.3.1 and 4.3.2 it is evident that an m - diagonal Hessian matrix of a twice continuously differentiable functional F can be numerically approximated and symmetrized by just $(m + 1)/2$ gradient evaluations at each point x at which g (x) is known. In

particular, the n × n full Hessian matrix can be numerically approximated and symmetrized with just n gradient evaluations.

# Chapter 5.

## Some Variations on a Least Squares Algorithm due to Gill and Murray.

### 5.1 Introduction

Gill and Murray (1975), (1976) have described a reliable and efficient method for the solution of the unconstrained least squares problem,

$$\underset{x \in R^n}{\text{Min}}\ F(x) = f(x)^T f(x), \qquad (5.1.1)$$

where $f : R^n \longrightarrow R^m$ $(m \geqslant n)$ and $F \in C^2(R^n)$. The method consists, eesentially of a judicious combination of the Gauss-Newton method and Newton's method, so that the second partial derivatives of the components $f_i$ $(1 \leqslant i \leqslant m)$ of f are required to be evaluated. A brief description of the method will now be given in order to clarify subsequent sections. A complete algorithm corresponding to one implementation of the method is given in Section 5.2.

If $F : R^n \longrightarrow R^1$ is defined as in (5.1.1), then Newton's method for the minimization of F consists of generating the sequence $(x^{(k)})$ from

$$x^{(k+1)} = x^{(k)} + p^{(k)} \qquad (k \geqslant 0), \qquad (5.1.2)$$

where $p^{(k)}$ is obtained by solving

$$G^{(k)}\ p^{(k)} = -g^{(k)} \qquad (5.1.3)$$

in which the $n \times n$ matrix $G^{(k)}$ and the $n \times 1$ vector $g^{(k)}$ are defined by

$$G^{(k)} = A^{(k)T} A^{(k)} + \sum_{i=1}^{m} f_i^{(k)} G_i^{(k)} \qquad (5.1.4)$$

and

$$g^{(k)} = A^{(k)T} f^{(k)}, \qquad (5.1.5)$$

and the $m \times n$ Jacobian matrix $A^{(k)}$ of $f$ at $x^{(k)}$ and the $n \times n$ Hessian matrix $G_l^{(k)}$ of $f_l$ at $x^{(k)}$ are defined by

$$A^{(k)} = \left( \partial_j f_i (x^{(k)}) \right)_{m \times n},$$

and

$$G_l^{(k)} = \left( \partial_j \partial_i f_l (x^{(k)}) \right)_{n \times n} \quad (l = 1, 2, \ldots, m),$$

where

$$\partial_j f_i (x) = \frac{\partial}{\partial x_j} f_i (x) \quad (i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n),$$

and

$$\partial_j \partial_i f_l (x) = \frac{\partial^2}{\partial x_j \partial x_i} f_l (x) \quad (i, j = 1, 2, \ldots, n;\ l = 1, 2, \ldots, m).$$

The Gauss-Newton method for the minimization of $F$ consists of generating the sequence $(x^{(k)})$ from (5.1.2) where $p^{(k)}$ is obtained by solving not (5.1.3) but

$$A^{(k)T} A^{(k)} p^{(k)} = - g^{(k)}. \qquad (5.1.6)$$

Thus the Gauss-Newton Method is derived from the Newton Method by neglecting the term $B^{(k)}$ in (5.1.4) defined by

$$B^{(k)} = \sum_{i=1}^{m} f_i^{(k)} G_i^{(k)}. \qquad (5.1.7)$$

The Gauss-Newton method is suitable for those problems in which

$\|B^{(k)}\|$ is small compared with $\|A^{(k)T} A^{(k)}\|$ for each $k \geqslant 0$. This situation arises in those data-fitting problems for which, if $x^*$ is a minimizer of $F$, then $F(x^*)$ is very small in magnitude. Indeed, if both the Newton and Gauss-Newton methods converge to $x^*$ and $F(x^*) = 0$ then usually both methods ultimately converge at the same rate.

In practice, the Newton and Gauss-Newton sequences generated from (5.1.2) frequently fail to converge to a minimizer $x^*$ of $F$ if $x^{(0)}$ is a poor estimate of $x^*$. The reliability of both the Newton and Guass-Newton methods is greatly improved if $(x^{(k)})$ is generated from

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)} \qquad (5.1.8)$$

where $\alpha^{(k)}$ is determined by using Algorithm 1.2.2.

If $\|B^{(k)}\|$ is small compared with $\|A^{(k)T} A^{(k)}\|$ and $A^{(k)}$ has rank n then $A^{(k)T} A^{(k)}$ is non-singular and the Gauss-Newton method would be expected to be effective. If, however, $A^{(k)}$ is rank-deficient or if $F^{(k)} / \|A^{(k)T} A^{(k)}\|$ is large then $A^{(k)T} A^{(k)}$ may not be an adequate approximation to $G^{(k)}$, and the Newton method would be expected to be effective. The Newton method involves the solution of

$$(A^T A + B)\, p = - A^T f , \qquad (5.1.9)$$

where the suffix k is suppressed for brevity. The least squares algorithm of Gill and Murray (1976) contains a procedure for solving (5.1.9). The procedure permits (5.1.9) to be solved when $A^T A + B$ is positive definite and when $A^T A + B$ is indefinite, and yields the Gauss-Newton vector $p_{GN}$ which satisfies

$$A^T A p = -g , \tag{5.1.10}$$

where A is not rank-deficient, as well as the Newton vector $p_N$ which satisfies (5.1.9).

The procedure for solving (5.1.9) depends upon the singular value decomposition

$$A = U \begin{bmatrix} S \\ \hline 0 \end{bmatrix} V^T \tag{5.1.11}$$

of A in which U is an $m \times m$ orthogonal matrix, V is an $n \times n$ orthogonal matrix, and $S = \text{Diag}\,(s_1, s_2, \ldots, s_n)$ is the matrix of singular values of A, with $s_{i+1} \leq s_i$ $(i = 1, 2, \ldots, n)$. For further details see the book by Lawson and Hanson (1974).

Because the columns of V provide a basis for $R^n$, there exists $z \in R^n$ such that

$$p = Vz . \tag{5.1.12}$$

It follows from (5.1.9), (5.1.11) and (5.1.12) that

$$(S^2 + V^T B V)z = -\left[S \;\vdots\; 0\right] U^T f. \tag{5.1.13}$$

As explained by Gill and Murray (1976), the numerical problems which arise when $S^2 + V^T B V$ is ill-conditioned can be avoided by computing the solution p of (5.1.9) as the sum of two components $p_1$, and $p_2$, where $p_1$ lies in the subspace of $R^n$ spanned by the columns of V corresponding to the 'large' singular values of A, and $p_2$ lies in the subspace spanned by the columns of V corresponding to the smaller singular values of A.

Let

$$S = \left[\begin{array}{c:c} S_1 & 0 \\ \hdashline 0 & S_2 \end{array}\right], \tag{5.1.14}$$

where $S_1 = \text{Diag}(s_1, s_2, \ldots, s_r)$ and $S_2 = \text{Diag}(s_{r+1}, \ldots, s_n)$. The integer r is referred to by Gill and Murray as the grade of A, and it is varied in a manner which depends upon the computational progress which has been made.

Let

$$V = \left[\begin{array}{c:c} V_1 & V_2 \end{array}\right], \tag{5.1.15}$$

where $V_1$ is an $n \times r$ matrix and $V_2$ is an $n \times (n - r)$ matrix.

If the solution p of (5.1.9) is written in the form

$$p = V_1 w + V_2 y, \tag{5.1.16}$$

where w is an $r \times 1$ vector and y is an $(n - r) \times 1$ vector, then by (5.1.13), (5.1.15) and (5.1.16), we have

$$S_1^2 w + V_1^T B V_1 w + V_1^T B V_2 y = -S_1 f_1 , \tag{5.1.17}$$

and

$$S_2^2 y + V_2^T B V_2 y + V_2^T B V_1 w = -S_2 f_2 , \tag{5.1.18}$$

where the $r \times 1$ vector $f_1$ and the $(n - r) \times 1$ vector $f_2$ are such that $f^T U = \left[\begin{array}{c:c:c} f_1^T & f_2^T & \tilde{f}^T \end{array}\right]$ and $\tilde{f}$ is an $(m - n) \times 1$ vector.

An approximation $\bar{w}$ to w may be obtained from (5.1.17) by neglecting the second and third terms on the left-hand side to obtain

$$\bar{w} = - S_1^{-1} f_1 \qquad (5.1.19)$$

and a corresponding approximation $\bar{y}$ to y is then obtained from (5.1.18) and satisfies

$$(S_2^2 + V_2^T BV_2) \bar{y} = - S_2 f_2 - V_2^T BV_1 \bar{w} . \qquad (5.1.20)$$

If $A^T A + B$ is positive definite then $S_2^2 + V_2^T BV_2$ is also positive definite, and $S_2^2 + V_2^T BV_2$ is not ill-conditioned because the "large" singular values of A are in $S_1$ . Therefore (5.1.20) may be solved for $\bar{y}$ by determining the Cholesky factorization of $S_2^2 + V_2^T BV_2$ .

If $A^T A + B$ is indefinite then Gill and Murray (1974) use their modified Newton-type method, in which the vector p satisfies

$$L D L^T p = - A^T f , \qquad (5.1.21)$$

where L is unit lower-triangular, D is diagonal, and $L D L^T$ is the modified Cholesky factorization of G where

$$G = A^T A + B , \qquad (5.1.22)$$

As described by Gill and Murray (1974a) L and D are such that

$$L D L^T = G + E , \qquad (5.1.23)$$

where E is a diagonal matrix, the elements of which are all zero if G is positive definite.

When G is indefinite, (5.1.20) may be solved for $\bar{y}$ by determining the modified Cholesky factorization of $S_2^2 + V_2^T B V_2$. This is equivalent to determining p by solving

$$(G + V_2 E V_2^T) p = - A^T f . \qquad (5.1.24)$$

This is not satisfactory for all values of r because if G has n negative eigenvalues then $x^T G x < 0$ $(\forall x \neq 0)$. Therefore there exists $q \in R^n$ such that $x = V_1 q$, and

$$x^T (G + V_2 E V_2^T) x = x^T G x < 0 ,$$

whence $G + V_2 E V_2^T$ is not positive definite. Therefore as a safeguard, p is recomputed with r = 0 if for a given positive number $\varepsilon$,

$$- g^T p < \varepsilon \| g \| \| p \| . \qquad (5.1.25)$$

If r = 0 then $V_1 = 0$, $S_1 = 0$, $V_2 = V$, and $S_2 = S$ so by (5.1.19) and (5.1.20), if $\bar{p} = V\bar{y}$ then $\bar{p}$ satisfies (5.1.21). As shown by Gill and Murray (1974a) there exists $\sigma > 0$ such that

$$- g^T p \geqslant \sigma \| g \| \| p \| . \qquad (5.1.26)$$

Therefore the safeguard ensures that the least squares algorithm of Gill and Murray is gradient-related, and Theorem 3.1.3 is applicable.

If for some k, $A^{(K)T} f^{(K)} = 0$ and $E^{(K)} \neq 0$ where $E^{(K)}$ corresponds to the modified Cholesky factorization of $G^{(K)}$, then $x^{(K)}$ is a saddle point of F. Gill and Murray (1974a) have shown that a descent direction $p^{(K)}$ from $x^{(K)}$ may be obtained by solving

$$L^{(K)T} y^{(K)} = e_j \quad , \tag{5.1.27}$$

where $e_j$ is column j of the n x n unit matrix and j is such that

$$d_j^{(K)} - E_j^{(K)} \leqslant d_i^{(K)} - E_i^{(K)} \quad (i = 1, 2, \ldots, n),$$

in which

$$D^{(K)} = \text{Diag} \left( d_1^{(K)}, \ldots, d_n^{(K)} \right),$$

$$E^{(K)} = \text{Diag} \left( E_1^{(K)}, \ldots, E_n^{(K)} \right),$$

and

$$L^{(K)} D^{(K)} L^{(K)T} = G^{(K)} + E^{(K)}.$$

The descent direction $p^{(K)}$ is then obtained from

$$p^{(K)} = V_2 \; y^{(K)}. \tag{5.1.28}$$

If analytical formulae for the $\partial_i \partial_j f_1(x)$ are not available, then, as explained by Gill and Murray (1976), row j of $V_2^{(K)T} B^{(K)}$ may be estimated by using the finite difference formula

$$v_j^{(K)T} B^{(K)} = f^{(K)T} \left\{ A \left( x^{(K)} + h \, v_j^{(K)} \right) - A \left( x^{(K)} \right) \right\} / h$$

$$(j = r + 1, \ldots, n) \tag{5.1.29}$$

in which h is a finite-difference steplength. If (5.1.29) is used to estimate $V_2^{(K)T} B^{(K)}$ then (n - r) evaluations of A are required. Now r is seldom very much less than n, so the use of (5.1.29) yields a very efficient method for estimating $V_2^{(K)T} B^{(K)}$.

Gill and Murray (1976) have also described a quasi-Newton method for estimating the $B^{(K)}$ when analytical formulae for the $\partial_i \partial_j f_1(x)$ are not available. The updating formula for $B^{(K)}$ which Gill and Murray propose is based upon the BFGS update and is given by

$$B^{(K+1)} = B^{(K)} + C^{(K)}, \tag{5.1.30}$$

where

$$C^{(K)} = \frac{y^{(K)} y^{(K)T}}{\alpha^{(K)} y^{(K)T} p^{(K)}} - \frac{W^{(K)} p^{(K)} p^{(K)T} W^{(K)T}}{p^{(K)T} W^{(K)} W^{(K)}}, \tag{5.1.31}$$

in which

$$W^{(K)} = A^{(K+1)T} A^{(K+1)} + B^{(K)} \tag{5.1.32}$$

and

$$y^{(K)} = A^{(K+1)T} A^{(K+1)} - A^{(K)T} f^{(K)}. \tag{5.1.33}$$

## 5.2. The Least Squares Algorithm of Gill and Murray.

An implementation of the least squares algorithm of Gill and Murray is available in the National Physical Laboratory Algorithms (NPL) library of programs and this implementation constitutes the foundation for the modifications which are described subsequently in this chapter. The algorithm corresponding to this implementation will be referred to as Algorithm 5.2.1 and is as follows.

Algorithm 5.2.1

It is assumed that $x^{(0)}$, m, n, $\varepsilon_i$ $(1 \leq i \leq 6)$, and $n_1$ are given.

Step 1. Set $k = 0$, $n_2 = n$, $n_4 = 1$, $n_5 = 0$, $n_6 = 0$, $n_7 = 1$, $\tau = 10$, and $n_9 = 2$.

Step 2. Compute $f^{(k)}$, $A^{(k)}$, $F^{(k)}$, and $g^{(k)}$ from $f^{(k)} = f(x^{(k)})$, $A^{(k)} = A(x^{(k)})$, $F^{(k)} = f^{(k)T} f^{(k)}$, and $g^{(k)} = A^{(k)T} f^{(k)}$.

Step 3. If $n_4 = 0$ then go to Step 9.

Step 4. Determine the singular value decomposition of $A^{(k)}$ according to

$$A^{(k)} = U^{(k)} \begin{bmatrix} S^{(k)} \\ \hline 0 \end{bmatrix} V^{(k)T},$$

where $S^{(k)} = \mathrm{Diag}(s_1^{(k)}, \ldots, s_n^{(k)})$, $V^{(k)} = \left[ V_1^{(k)} \mid \cdots \mid V_n^{(k)} \right]$, in which the singular values $s_i^{(k)}$ of $A^{(k)}$ are such that $s_{i+1}^{(k)} \leq s_i^{(k)}$ $(i = 1, \ldots, n-1)$.

Step 5. Determine the number j of singular values $s_i^{(k)}$ $(i \neq 1)$ such that $s_i^{(k)} \geq 10\, \varepsilon_2\, s_1^{(k)}$ and set $n_3 = 1 + j$.

Step 6. Set $n_8 = 0$.

Step 7. If $n_3 < n_2$ then set $n_2 = n_3$.

Step 8. If $\tau > 10^{-1}$ or ( $\tau > 10^{-2}$ and $n_5 = 0$) then go to Step 13.

Step 9. Set $n_8 = 1$.

Step 10. If $\tau > 10^{-2}$ then go to Step 13.

Step 11. If $n_5 = 0$ then determine $n_2$ such that

$$s_1^{(K)} / s_{n_2}^{(K)} + s_{n_2+1}^{(K)} / s_1^{(K)} \leqq s_1^{(K)} / s_j^{(K)} + s_{j+1}^{(K)} / s_1^{(K)}$$

$$(j = 1, \ldots, l - 1)$$

where $s_l^{(K)}$ is the last non zero singular value of $A^{(K)}$.

Step 12. If $n_5 = 1$ then determine $n'$ such that $n' < n_2$,

$s_{n'}^{(K)} \geqslant 10^{1+\nu} \cdot s_{n_2}^{(K)}$, and $s_{n'+1}^{(K)} < 10^{1+\nu} \cdot s_{n_2}^{(K)}$, where

$$\nu = \log_{10} s_{n_2}^{(K)} - \left[\log_{10} s_{n_2}^{(K)}\right] ,$$

in which $[u]$ is the largest integer which is not greater than u, and set $n_2 = n'$.

Step 13. Set $n_5 = n_8$ .

Step 14. If $n_5 = 0$ then set $n_2 = n_3$ .

Step 15. If $k \neq 0$, and

$$\| x^{(K)} - x^{(K-1)} \|_2 \leq ( \varepsilon_1 + \varepsilon_3 ) (1 + \| x^{(K)} \|_2 ) \quad \text{and}$$

$$| F^{(K)} - F^{(K-1)} | ( \varepsilon_1 + \varepsilon_3 )^2 (1 + F^{(K)} ) \quad \text{and}$$

$$\| g^{(K)} \|_2 \leq \varepsilon_1^{1/3} (1 + F^{(K)} )$$

or

$$\| g^{(K)} \|_2^2 < (F^{(K)})^{1/2} \varepsilon_1 \quad \text{or} \quad F^{(K)} < \varepsilon_1^2 ,$$

then set $n_7 = 0$.

Step 16. Compute $\gamma$ from $\gamma = F^{(K)2} / s_1^{(K)4}$.

Step 17. If $(n_6 = 0$ or $\gamma < \varepsilon_4 )$ and $n_7 = 0$, then stop. The iterate $x^{(K)}$ is an adequate estimate of a minimizer of F.

Step 18. If $n_2 = 0$ or $n_7 = 0$ then set $p_1^{(K)} = 0$ and go to Step 20.

Step 19. Compute $p_1^{(K)}$ from $p_1^{(K)} = - V_1^{(K)} S_1^{(K)-1} f_1^{(K)}$,

where $V_1^{(K)} = \left[ v_1^{(K)} \mid - - - \mid v_{n_2}^{(K)} \right]$, $S_1^{(K)} = \text{Diag} ( s_1^{(K)}, \ldots, s_{n_2}^{(K)} )$,

and $f^{(K)T} U^{(K)} = \left[ f_1^{(K)T} \mid f_2^{(K)T} \mid \tilde{f}^{(K)T} \right]$.

Step 20. If $n_5 = 0$ then set $p_2^{(K)} = 0$ and go to Step 32.

Step 21. If $n_1 = 0$ then go to Step 23.

Step 22. Estimate $T^{(K)} = V_2^{(K)T} B^{(K)}$ by using the finite-difference formula (5.1.29), and compute $q^{(K)T} = T^{(K)} p_1^{(K)}$,

$$Q^{(K)} = T^{(K)} V_2^{(K)}, \text{ where } V_2^{(K)} = \left[ v_{n_2+1}^{(K)} \,\vdots\, \cdots \,\vdots\, v_n^{(K)} \right]$$

and then go to Step 26.

Step 23. If $n_9 = 1$ or $n_4 = 1$ then compute $B^{(K)}$ defined by (5.1.7) analytically.

Step 24. Set $n_9 = 2$.

Step 25. Compute $T^{(K)} = V_2^{(K)T} B^{(K)}$, $Q^{(K)} = T^{(K)} V_2^{(K)}$ and $q^{(K)} = T^{(K)} p_1^{(K)}$

Step 26. Determine the modified Cholesky factorization of $S_2^{(K)2} + Q^{(K)}$ according to

$$L^{(K)} D^{(K)} L^{(K)T} = S_2^{(K)2} + Q^{(K)} + E^{(K)}$$

Step 27. Set $n_6 = 0$.

Step 28. If $\|E^{(K)}\|_\infty \neq 0$ then set $n_6 = 1$.

Step 29. If $n_6 = 0$ or $n_7 = 1$ then determine $y^{(K)}$ from

$$L^{(K)} D^{(K)} L^{(K)T} y^{(K)} = - S_2^{(K)} f_2^{(K)} - q^{(K)}$$

where $S_2^{(K)} = \text{Diag}\,(s_{n_2+1}^{(K)}, \ldots, s_n^{(K)})$ and $f_2^{(K)}$ is defined as in Step 19.

Step 30. If $n_6 = 1$ and $n_7 = 0$ then determine $y^{(K)}$ from (5.1.27).

Step 31. Compute $p_2^{(K)}$ from $p_2^{(K)} = V_2^{(K)} y^{(K)}$.

Step 32. Compute $p^{(K)}$ from $p^{(K)} = p_1^{(K)} + p_2^{(K)}$

Step 33. Set $n_4 = 1$.

Step 34. If $g^{(K)T} p^{(K)} > 0$ then set $p^{(K)} = -p^{(K)}$.

Step 35. If $-g^{(K)T} p^{(K)} \geqslant \varepsilon_5 \|g^{(K)}\|_2 \cdot \|p^{(K)}\|_2$ or $n_2 = 0$, then go to Step 38.

Step 36. If $n_5 = 0$ then set $n_5 = 1$ and $n_9 = 1$.

Step 37. Set $n_2 = 0$, $n_4 = 0$, and go to Step 9.

Step 38. Determine $\alpha^{(K)}$ by using Algorithm 1.2.2.
If Algorithm 1.2.2 could successfully determine $\alpha^{(K)}$ then $f^{(K+1)}$, $F^{(K+1)}$, $A^{(K+1)}$ are determined by Algorithm 1.2.2, where $x^{(K+1)} = x^{(K)} + \alpha^{(K)} p^{(K)}$, $f^{(K+1)} = f(x^{(K+1)})$, $F^{(K+1)} = f^{(K+1)T} f^{(K+1)}$, $A^{(K+1)} = A(x^{(K+1)})$

Step 39. Compute $\tau$ from

$$\tau = (F^{(k)} - F^{(k+1)}) / F^{(k)}$$

If Algorithm 1.2.2 cannot determine $\alpha^{(k)}$ then the value of $\tau$ will be zero.

Step 40. If $\tau \neq 0$ then set k = k + 1 and go to Step 3.

Step 41. Set $n_4 = 0$.

Step 42. If $n_5 = 0$ the set $n_5 = 1$ and $n_9 = 1$.

Step 43. If $n_2 > 0$ then set k = k + 1 and go to Step 3.

Step 44. Stop. □

In the original implementation of Algorithm 5.2.1, Step 23 is as follows.

Step 23. If $n_4 = 1$ then compute $B^{(k)}$ defined by (5.1.7) analytically.

Also, in the original implementation of Algorithm 5.2.1, Step 42 does not exist. The original NPL program failed to solve problems 27 and 28 of Appendix 2, but the program based upon Algorithm 5.2.1 in this section decreased the sum to 1.90 from $10^{25}$.

In the original implementation of Algorithm 5.2.1, the test in Step 35 is missing although Gill and Murray (1976) mention it in that report.

The significance of the integers $n_1 - n_9$ is as follows.

If $n_1 = 0$ then $B^{(k)}$ is computed by using analytical formulae for $\partial_i \partial_j f_1(x)$ ; if $n_1 = 1$ the finite-difference formula (5.1.29) is used. The integers $n_2$ and $n_3$ are the grade of $A^{(k)}$ and the current number of 'large' elements of $S^{(k)}$ respectively. If $n_4 = 0$ then the steplength algorithm (Step 38) has not been successful. If $n_5 = 0$ then a Gauss-Newton step has been taken, while if $n_5 = 1$ then a graded Gauss-Newton step or a Newton step has been taken. If $n_6 = 0$ then $S_2^{(k)2} + V_2^{(k)T} B^{(k)} V_2^{(k)T}$ is positive definite, while if $n_6 = 1$ then it is not positive-definite. If $n_7 = 0$ then $x^{(k)}$ is 'close' to $x^*$ in the sense that $x^{(k)}$ satisfies the convergence criteria in Step 15. If $n_8 = 0$ then a Gauss-Newton step must be taken, while if $n_8 = 1$ then a graded Gauss-Newton step must be taken. If $n_9 = 1$ then $B^{(k)}$ must be calculated, while if $n_9 \neq 1$ then $B^{(k)}$ need not be calculated.

## 5.3 An Alternative Graded Gauss-Newton Search Direction.

The modified Gauss-Newton search direction p used by Gill and Murray is approximated by using (5.1.16), (5.1.19), and (5.1.20). In order to find a more accurate approximation to p, Gill and Murray apply the following algorithm.

Algorithm 5.3.1

Step 1. Set $\bar{p}^{(0)} = 0$.

Step 2. For $j = 0, 1, \ldots$ compute $\bar{w}^{(j+1)}$ and $\bar{y}^{(j+1)}$ such that

$$S_1^2 \bar{w}^{(j+1)} = -S_1 f_1 - V_1^T B \bar{p}^{(j)} \qquad (5.3.1)$$

$$\text{and } (S_2^2 + V_2^T BV_2)\,\bar{y}^{(j+1)} = -S_2 f_2 - V_2^T BV_1\,\bar{w}^{(j+1)}. \tag{5.3.2}$$

Step 3. Set $\bar{p}^{(j+1)} = V_1\,\bar{w}^{(j+1)} + V_2\,\bar{y}^{(j+1)}$. □

The vectors $\bar{w}$ and $\bar{y}$ defined by (5.1.19) and (5.1.20) are determined from Algorithm 5.3.1 according to

$$\bar{w} = \bar{w}^{(1)} \text{ and } \bar{y} = \bar{y}^{(1)}.$$

Therefore, at iteration 1, Algorithm 5.3.1 determines

$$\bar{p}^{(1)} = V_1\,\bar{w}^{(1)} + V_2\,\bar{y}^{(1)}$$

$$= V_1\,\bar{w} + V_2\,\bar{y} \tag{5.3.3}$$

as the first approximation to p.

The following theorem indicates that the sequence $\bar{p}^{(k)}$ generated from Algorithm 5.3.1 will converge.

Theorem 5.3.1

If $\bar{p}^{(j)}$ and $\bar{p}^{(j+1)}$ are two approximations to p obtained by using Algorithm 5.3.1, then their relative errors satisfy the inequality

$$\|p - \bar{p}^{(j+1)}\| \;/\; \|p\| \leq \sigma\varepsilon(1 + \varepsilon\gamma)\,\|p - \bar{p}^{(j)}\| \;/\; \|p\|$$

where

$$\varepsilon = \|B\|, \quad \sigma = \|S_1^{-2}\|, \text{ and } \gamma = \|(S_2^2 + V_2^T BV_2)^{-1}\|$$

Proof.

See Gill and Murray (1976). □

In practice, usually, the first iteration of Algorithm 5.3.1 will suffice. Therefore, the approximated search direction will be given by $\bar{p} = \bar{p}^{(1)}$.

Therefore by (5.3.3)

$$\bar{p} = V_1 \bar{w} + V_2 \bar{y}, \tag{5.3.4}$$

where

$$\bar{w} = - S_1^{-1} f_1 , \tag{5.3.5}$$

and

$$M\bar{y} = - S_2 f_2 - V_2^T BV_1 \bar{w} , \tag{5.3.6}$$

in which

$$M = S_2^2 + V_2^T BV_2 \tag{5.3.7}$$

is an $(n - r) \times (n - r)$ symmetric matrix, and r is the grade of A as explained in section (5.1.1).

When $r = n$, then $V_2 = 0$, $S_2 = 0$, $S_1 = S$, $V_1 = V$ and therefore

$$\bar{p} = - (A^T A)^{-1} A^T f$$

is the Gauss-Newton direction.

Similarly, when $r = 0$, $\bar{p}$ is the Newton direction obtained from (5.1.9).

But when $r \neq 0$ and $r \neq n$, then $\bar{p} = \bar{p}^{(1)}$ is called by Gill and Murray (1976), the graded Gauss-Newton search direction.

The following theorem will be used to construct a method for determining an alternative graded Gauss-Newton search direction.

Theorem 5.3.2

Let $\bar{p}$ be defined by (5.3.4) - (5.3.7). Then

$$\bar{p} = -\bar{G}^{-1} g, \tag{5.3.8}$$

where

$$g = A^T f,$$

and

$$\bar{G} = V \begin{bmatrix} S_1^2 & 0 \\ N & M \end{bmatrix} V^T \tag{5.3.9}$$

in which $N = V_2^T B V_1$, $V_1$ and $V_2$ are defined as in (5.1.15).

Proof.

Without loss of generality, we assume that M is positive definite, since otherwise, to solve (5.3.6), Gill and Murray apply the modified Cholesky factorization and M is replaced with a positive definite matrix $\bar{M}$. Now it can be shown that $\bar{G}$ is non-singular and $\bar{G}^{-1} = H$, where

$$H = V \begin{bmatrix} S_1^{-2} & 0 \\ -\bar{M}^{-1} N S_1^{-2} & \bar{M}^{-1} \end{bmatrix} V^T, \tag{5.3.10}$$

because, by orthogonality of V, we have

$$H\bar{G} = V \begin{bmatrix} I_r & 0 \\ 0 & I_{n-r} \end{bmatrix} V^T = I_n$$

and

$$\bar{G}H = I_n.$$

By (5.1.11), (5.1.15), we have

$$g = A^T f$$

$$= V \left[ S \mid 0 \right] U^T f$$

$$= V_1 S_1 f_1 + V_2 S_2 f_2$$

$$= V \begin{bmatrix} S_1 f_1 \\ S_2 f_2 \end{bmatrix} .$$

From (5.3.4) - (5.3.7) we have

$$\bar{p} = -V_1 S_1^{-1} f_1 - V_2 M^{-1} (S_2 f_2 - N S_1^{-1} f_1)$$

$$= - V \begin{bmatrix} S_1^{-2} & 0 \\ -M^{-1} N S_1^{-2} & M^{-1} \end{bmatrix} V^T V \begin{bmatrix} S_1 f_1 \\ S_2 f_2 \end{bmatrix} . \qquad (5.3.11)$$

Therefore, from (5.3.10) and (5.3.11) we have

$$\bar{p} = - \bar{G}^{-1} g. \quad \square$$

Corollary 5.3.1

Suppose that the hypotheses of Theorem 5.3.2 are valid.

(a) If $r = n$ then $\bar{G} = A^T A$ ;

(b) If $r = 0$ then $\bar{G} = A^T A + B$;

(c) If $0 < r < n$ then $\bar{G} = A^T A + V_2 V_2^T B$ .

Proof.

If $r = n$, then $S_1 = S$, $S_2 = 0$, $V_1 = V$, $V_2 = 0$.

Therefore, $N = 0$, and $M = 0$. Thus (a) holds.

If $r = 0$, then $V_1 = 0$, $S_1 = 0$, $S_2 = S$, $V_2 = V$, $N = 0$,

and $M = S^2 + V^T BV$. Therefore (b) holds.

Suppose that $0 < r < n$. Then, by (5.3.9), (5.1.15), and orthogonality of V, we have

$$\bar{G} = \left[ V_1 \mid V_2 \right] \left[ \begin{array}{c|c} S_1^2 & 0 \\ \hline N & M \end{array} \right] \left[ \begin{array}{c} V_1^T \\ \hline V_2^T \end{array} \right]$$

$$= V_1 S_1^2 V_1^T + V_2 N V_1^T + V_2 M V_2^T$$

$$= A^T A + V_2 V_2^T B . \quad \square$$

The following theorem gives an alternative graded Gauss-Newton search direction.

Theorem 5.3.3

If 1. M is defined by (5.3.7) and is positive definite;

2. $\bar{w}$ is defined by (5.3.5);

3. $\bar{\bar{p}} = V_1 \bar{w} + V_2 \bar{\bar{y}}$, where $\bar{\bar{y}} = - M^{-1} S_2 f_2$ ,

then

$$\bar{\bar{p}} = -\bar{\bar{G}}^{-1} g, \tag{5.3.14}$$

where $g = A^T f$, and

$$\bar{\bar{G}} = V \begin{bmatrix} S_1^2 & 0 \\ 0 & M \end{bmatrix} V^T . \tag{5.3.15}$$

Proof

Since $S_1^2$ is a positive definite diagonal matrix and M is a positive definite matrix, then $\bar{\bar{G}}$ defined by (5.3.15) is positive definite and therefore, $\bar{\bar{G}}^{-1}$ exists. Moreover, it is easily verified that

$$\bar{\bar{G}}^{-1} = V \begin{bmatrix} S_1^{-2} & 0 \\ 0 & M^{-1} \end{bmatrix} V^T . \tag{5.3.16}$$

Now by (5.3.14) and Hypothesis 3 we have

$$\bar{\bar{p}} = -V_1 S_1^{-1} f_1 - V_2 M^{-1} S_2 f_2$$

$$= -\begin{bmatrix} V_1 & V_2 \end{bmatrix} \begin{bmatrix} S_1^{-1} & f_1 \\ M^{-1} S_2 & f_2 \end{bmatrix}$$

$$= -\begin{bmatrix} V_1 & V_2 \end{bmatrix} \begin{bmatrix} S_1^{-2} & 0 \\ 0 & M^{-1} \end{bmatrix} V^T V \begin{bmatrix} S_1 & f_1 \\ S_2 & f_2 \end{bmatrix}. \tag{5.3.17}$$

By (5.3.16) (5.3.14) we have

$$\bar{\bar{p}} = -\bar{\bar{G}}^{-1} V \begin{bmatrix} S_1 & f_1 \\ S_2 & f_2 \end{bmatrix}$$

$$= -\bar{\bar{G}}^{-1} g,$$

since $A^T f = V \begin{bmatrix} S_1 & f_1 \\ S_2 & f_2 \end{bmatrix}$ . □

Corollary 5.3.2

Suppose that the hypotheses of Theorem 5.3.2 are valid.

(a) If $r = n$ then $\bar{\bar{G}} = A^T A$.

(b) If $r = 0$ then $\bar{\bar{G}} = A^T A + B$.

(c) If $0 < r < n$ then $G = A^T A + V_2 V_2^T B V_2 V_2^T$ .

Proof.

If $r = n$, then $M = 0$, and therefore (a) holds.

Suppose that $r < n$. Then by (5.3.15) we have

$$\bar{\bar{G}} = V \left[\begin{array}{c|c} S_1^2 & 0 \\ \hline 0 & M \end{array}\right] V^T$$

$$= V_1 S_1^2 V_1^T + V_2 M V_2^T$$

$$= A^T A + V_2 V_2^T B V_2 V_2^T .$$

If $r = 0$ then, $V_2 = V$. Thus (b) holds; otherwise (c) holds. □

It would appear that $\bar{\bar{p}}$ determined by Theorem 5.3.3 is a more satisfactory choice of p than $\bar{p}$ given by (5.3.4)-(5.3.7), because if

M is positive definite, then $\bar{\bar{G}}$ defined by (5.3.15) is positive definite, while $\bar{G}$ defined by (5.3.9) is merely non-singular.

Algorithm 5.3.2 corresponds to the search direction $\bar{\bar{p}}$. Algorithm 5.3.2 is the same as Algorithm 5.2.1 save that Step 29 is replaced with Step 29′, which is as follows.

Step 29′. If $n_6 = 0$ or $n_7 = 1$ then determine $y^{(k)}$ from

$$L^{(k)} D^{(k)} L^{(k)T} y^{(k)} = - S_2^{(k)} f_2^{(k)} .$$

Also, the computation of $q^{(k)}$ at steps 22 and 25 has to be omitted. Moreover, Algorithm 5.3.3 is defined to be the same as Algorithm 5.3.2, save that $n_1 = 1$. That is, B is numerically approximated by using finite-differences.

## 5.4 The First Modification of Algorithm 5.3.2

A minimizer $x^*$ of $F : R^n \longrightarrow R^1$ defined by (5.1.1) is a solution of the equation

$$g(x) = 0, \tag{5.4.1}$$

where

$$g(x) = A(x)^T f(x). \tag{5.4.2}$$

The application of Newton's method for the solution of the nonlinear equations (5.4.1) yields the sequence $(x^{(k)})$ defined by (2.1.6) with

$$g'(x^{(k)}) = A(x^{(k)})^T A(x^{(k)}) + B^{(k)}, \tag{5.4.3}$$

where $B^{(K)}$ is as in (5.1.7).

In Chapter 2, some of the iterative methods for the solution of (5.4.1) which have higher ultimate rate of convergence than Newton's method have been discussed, in addition to Newton's method. Wolfe (1976) applied the iterative procedures (2.3.13), (2.3.14), (2.3.15), and (2.3.13), (2.3.16) and (2.3.17) with p = 2 to solve (5.4.1) to improve the performances of a least squares algorithm due to Meyer and Roth (1972). In this case, he set U (x) required in (2.3.14) and (2.3.16) to be

$$U(x) = A(x)^T A(x) + \lambda D(x) \quad (\forall x \in R^n), \qquad (5.4.4)$$

where $$D(x) = \text{Diag}(d_1(x), \ldots, d_n(x)), \qquad (5.4.5)$$

in which $d_i(x) = (A(x)^T A(x))_{ii} \quad (1 \leqslant i \leqslant n)$

and $\lambda$ is a scalar which varies from iteration to iteration. Also, Wolfe (1976) observed that the iterative method defined by (2.3.13) (2.3.16), and (2.3.17) is more computationally efficient than the one defined by (2.3.13) - (2.3.15). These observations and Theorem 2.3.3 give rise to the conjecture that the rate of convergence and the computational efficiency of Algorithm 5.3.2 may be improved by replacing the Newton method, upon which it is based, with the iterative procedure (2.3.13), (2.3.16) and (2.3.17) in which U (x) is defined by

$$U(x) = \bar{\bar{G}}(x) \quad (\forall x \in R^n),$$

where $\bar{\bar{G}}(x)$ is defined as in Theorem (5.3.3); thus Algorithm 5.4.1 is obtained from Algorithm 5.3.2 by modifying Step 35 according to

Step 35'. If $(-g^{(K)T} p^{(K)}) \geqslant \varepsilon_5 \|g^{(K)}\|_2 \|p^{(K)}\|_2$

or

$n_2 = 0$, then go to Step 2.1,

and appending the following steps.

Step 2.1. If $n_4 = 0$ or ($n_6 = 1$ and $n_7 = 0$) or $\|p^{(K)}\|_2 > \varepsilon_6$ then go to Step 38.

Step 2.2. Set $\bar{x}^{(K)} = x^{(K)} + p^{(K)}$ and $j = 1$.

Step 2.3. Compute $\bar{f}^{(K)} = f(\bar{x}^{(K)})$ and $\bar{\bar{F}}^{(K)} = F(\bar{x}^{(K)})$.

Step 2.4. If $\bar{\bar{F}}^{(K)} \leqslant F^{(K)} + \beta\, g^{(K)T} p^{(K)}$ then go to Step 2.6.

Step 2.5. Go to Step 38, making use of $\bar{f}^{(K)}$ and $\bar{\bar{F}}^{(K)}$ in the Algorithm 1.2.2 if required.

Step 2.6. If $n_2 = 0$ or $n_7 = 0$ then go to Step 2.9.

Step 2.7. Compute $\bar{p}_1^{(K)}$ from

$$\bar{p}_1^{(K)} = -V_1^{(K)} S_1^{(K)-1} \bar{f}_1^{(K)}$$

Step 2.8. Set $\bar{p}_2^{(K)} = 0$ and go to Step 2.10.

Step 2.9. Set $\bar{p}_1^{(k)} = 0$.

Step 2.10. If $n_5 = 0$ then go to Step 2.13.

Step 2.11. Compute $\bar{q}^{(k)} = T^{(k)} \bar{p}_1^{(k)}$

Step 2.12. Determine $\bar{p}_2^{(k)}$ from

$$L^{(k)} D^{(k)} L^{(k)T} \bar{y}^{(k)} = - S_2^{(k)} \bar{f}_2^{(k)} - \bar{q}^{(k)}$$

and

$$\bar{p}_2^{(k)} = V_2^{(k)} \bar{y}^{(k)}$$

Step 2.13. Set $\bar{p}^{(k)} = \bar{p}_1^{(k)} + \bar{p}_2^{(k)}$

Step 2.14. If $g^{(k)T} \bar{p}^{(k)} < 0$ and $\|\bar{p}^{(k)}\|_2 \leq \varepsilon_6$ and ($n_2 = 0$ or $- g^{(k)T} \bar{p}^{(k)} \geqslant \varepsilon_5 \|g^{(k)}\|_2 \|\bar{p}^{(k)}\|_2$ ) then go to Step 2.18.

Step 2.15. Set $p^{(k+1)} = \bar{x}^{(k)} - x^{(k)}$, $x^{(k+1)} = \bar{x}^{(k)}$, $F^{(k+1)} = \bar{F}^{(k)}$, $f^{(k+1)} = \bar{f}^{(k)}$, and compute $\tau = (F^{(k)} - F^{(k+1)}) / F^{(k)}$

Step 2.16. Compute $A^{(k+1)} = A(x^{(k+1)})$ and $g^{(k+1)} = A^{(k+1)T} f^{(k+1)}$

Step 2.17. Set $k = k + 1$ and go to Step 3.

Step 2.18. Set $\tilde{x}^{(k)} = \bar{x}^{(k)} + \bar{p}^{(k)}$

Step 2.19. Compute $\tilde{f}^{(k)} = f(\tilde{x}^{(k)})$ and $\tilde{F}^{(k)} = \tilde{f}^{(k)T} \tilde{f}^{(k)}$

Step 2.20. If $\tilde{F}^{(K)} \leqslant \bar{F}^{(K)} + \beta\, g^{(K)T} \bar{p}^{(K)}$ then go to Step 2.22.

Step 2.21. Go to Step 2.15.

Step 2.22. Set $j = j + 1$, $\bar{F}^{(K)} = \tilde{F}^{(K)}$, $\bar{x}^{(K)} = \tilde{x}^{(K)}$, $\bar{f}^{(K)} = \tilde{f}^{(K)}$.

Step 2.23. If $j = p$ then go to Step 2.15.

Step 2.24. Go to Step 2.6. □

In order to determine sufficient conditions for the convergence of Algorithm 5.4.1 the following lemma is required. The sequences appearing in the following lemma are generated by Algorithm 5.4.1.

Lemma 5.4.1

If 1. sequences $(A^{(K)})$, $(B^{(K)})$ are bounded;

2. $\exists\, \sigma_1 > 0$ such that

$$s_i^{(K)2} \geqslant \sigma_1 \quad (\forall i, \text{ if } s_i^{(K)} \neq 0\,,\ \forall k \geqslant 0);$$

3. $\bar{M}^{(K)}$ is obtained from modified Cholesky factorization of $M^{(K)}$ as in Step 26, where

$$M^{(K)} = S_2^{(K)2} + V_2^{(K)T} B^{(K)} V_2^{(K)};$$

4. $\bar{\bar{G}}^{(K)}$ is defined by

$$\bar{\bar{G}}^{(K)} = V^{(K)} \begin{bmatrix} S_1^{(K)2} & 0 \\ 0 & \bar{M}^{(K)} \end{bmatrix} V^{(K)T}$$

then $\exists \gamma > 0$ such that

$$x^T \bar{\bar{G}}^{(k)} x \geq \gamma \| x \|_2^2 \qquad (\forall x \in R^n) .$$

Proof

By Hypothesis 1 and orthogonality of $V^{(k)} = \left[ V_1^{(k)} \mid V_2^{(k)} \right]$, the sequence $(M^{(k)})$ is bounded. Therefore by Theorem 3.1.2, $\exists \sigma_2 > 0$ such that

$$y^T \bar{M}^{(k)} y \geq \sigma_2 \| y \|_2^2 \quad (\forall y \in R^{n-r}). \tag{5.4.6}$$

Now, for all $x \in R^n$, from Hypotheses 4 we have

$$x^T \bar{\bar{G}}^{(k)} x = y^T \left[ \begin{array}{c|c} S_1^{(k)2} & 0 \\ \hline 0 & \bar{M}^{(k)} \end{array} \right] y , \tag{5.4.7}$$

where $y^{(k)} = V^{(k)T} x$ .

Let

$$y^{(k)} = \left[ \begin{array}{c} y_1^{(k)} \\ \hline y_2^{(k)} \end{array} \right] ,$$

where $y_1^{(k)} \in R^r$ and $y_2^{(k)} \in R^{n-r}$ .

Therefore, from (5.4.6) and (5.4.7) and Hypothesis 2 we have

$$\begin{aligned} x^T \bar{\bar{G}}^{(k)} x &= y_1^{(k)T} S_1^{(k)2} y_1^{(k)} + y_2^{(k)T} \bar{M}^{(k)} y_2^{(k)} \\ &\geq \sigma_1 \| y_1^{(k)} \|_2^2 + \sigma_2 \| y^{(k)} \|_2^2 \\ &\geq \gamma \| y^{(k)} \|_2^2 , \end{aligned} \tag{5.4.8}$$

where

$$\gamma = \min\ (\sigma_1 ,\ \sigma_2)\ .$$

Since $V^{(K)}$ is orthogonal then $\| x \|_2^2 = \| y^{(K)} \|_2^2$ ; thus from (5.4.8) the proof is completed. □

Theorem 5.4.1

If 1. $f : D \subset R^n \longrightarrow R^m$ is twice continuously differentiable $(\forall\, x \in D)$;

2. the set $S \subset D$ of critical points of $F\ (= f^T f)$ is finite;

3. $x^{(0)} \in D$ is such that the closure $\bar{\Omega}\,(F^{(0)})$ of $\Omega\,(F^{(0)})$ is compact, where

$$\Omega\,(F^{(0)}) = \left\{ x \in D : F\,(x) \leq F^{(0)} \right\},$$

$CO\left[\bar{\Omega}\,(F^{(0)})\right] \subset D$, where $CO\left[\bar{\Omega}\,(F^{(0)})\right]$ is the closed convex hull of $\bar{\Omega}\,(F^{(0)})$;

4. $\exists\ \rho_1, \rho_2 > 0$ such that

$$\| B\,(x) \| \leq \rho_1 \quad , \qquad (\forall\, x \in \bar{\Omega}(F^{(0)}));$$

$$\| A\,(x) \| \leq \rho_2 \quad , \qquad (\forall\, x \in \bar{\Omega}(F^{(0)})),$$

where

$$A(x) = \left( \partial_i f_j (x) \right)_{m \times n},$$

$$B(x) = \sum_{l=1}^{m} f_l \, G_l (x),$$

in which $G_l(x) = \left( \partial_i \partial_j f_l (x) \right)_{n \times n}$ $(1 \leqslant l \leqslant m)$,

then the sequence $(x^{(k)})$ generated from Algorithm 5.4.1 with Step 15 omitted is such that $x^{(k)} \longrightarrow x^*$ (as $k \longrightarrow \infty$); where $x^* \in S$.

Proof.

The proof, in which Lemma 5.4.1 may be used, is similar to that of Theorem 3.2.1 and will therefore be omitted. □

NOTE:

If the direction of search defined by Gill and Murray is used then Theorem 5.4.1 is not valid because $\bar{G}$ defined by (5.3.9) is not positive definite even if M is replaced with $\bar{M}$ where

$$\bar{M} = L D L^T$$

$$= M + E$$

as in step 26 of Algorithm 5.2.1. Thus Lemma 5.4.1 would not be valid, and therefore cannot be used to prove Theorem 5.4.1.

## 5.5. The Second Modification of Algorithm 5.3.2.

Theorem 2.3.4 with $X = R^n$, and $Y = R^n$ may be used to construct a least squares algorithm in order to economize on the evaluations of f, A and B. The following observations give rise to a modification of Algorithm 5.3.2 which generates a sequence $(x^{(k)})$, the convergence

of which is guaranteed by Theorem 2.3.4.

The Jacobian matrix $A(x)$ of $f$ at $x$ is such that

$$A(x)^T = \left[ a_1(x) \mid \cdots \mid a_m(x) \right], \tag{5.5.1}$$

where

$$a_l(x) = (\partial_1 f_l(x), \ldots, \partial_n f_l(x))^T \quad (l = 1, 2, \ldots, m) \tag{5.5.2}$$

For each $x, \bar{x} \in R^n$, let the $m \times n$ matrix $\bar{A}$ be defined by

$$\bar{A}^T = \left[ \bar{a}_1 \mid \cdots \mid \bar{a}_m \right], \tag{5.5.3}$$

where

$$\bar{a}_l = a_l(x) + G_l(x)(\bar{x} - x) \quad (l = 1, 2, \ldots, m), \tag{5.5.4}$$

and let $\bar{f} \in R^m$ be defined by

$$\bar{f}_l = f_l(x) + a_l(x)^T (\bar{x} - x) + \tfrac{1}{2} (\bar{x} - x)^T G_l(x) (\bar{x} - x) \quad (l = 1, 2, \ldots, m). \tag{5.5.5}$$

Let $U : R^n \longrightarrow L(R^n)$ be defined by

$$U(\bar{x}) = \bar{A}^T \bar{A} + \sum_{i=1}^{m} \bar{f}_i G_i(x), \tag{5.5.6}$$

where

$$\bar{x} = x - G(x)^{-1} g(x) \tag{5.5.7}$$

in which $g : R^n \longrightarrow R^n$ is defined by (5.4.2) and

$$G(x) = A(x)^T A(x) + \sum_{i=1}^{m} f_i(x)\, G_i(x). \qquad (5.5.8)$$

Let $T : R^n \longrightarrow R^n$ be defined by

$$T(\bar{x}) = \bar{A}^T \bar{f}. \qquad (5.5.9)$$

Suppose that $f \in C^3(B(x^*, r))$, and that $G(x)^{-1}$ exists in $B(x^*, r)$, where $x^*$ is a solution of (5.1.1). Then by Taylor's theorem, $\exists M, N > 0$ such that

$$\|U(\bar{x}) - G(\bar{x})\| \leq M \|x - x^*\| \quad (\forall x \in B(x^*, r)),$$

where $\bar{x}$ is given by (5.5.7), and

$$\|T(\bar{x}) - g(\bar{x})\| \leq N \|x - x^*\|^p \quad (\forall x \in B(x^*, r))$$

in which $p = 2$ if $F(x^*) \neq 0$ and $p = 3$ if $F(x^*) = 0$.
Therefore if $r > 0$ is sufficiently small, Theorem 2.3.4 guarantees that the sequence $(x^{(k)})$ generated from (2.3.18) and (2.3.19) with T and U defined by (5.5.9) and (5.5.6) converges to the solution $x^*$ of (5.1.1) and that (2.3.21) holds with $p = 2$ if $F(x^*) \neq 0$ and $p = 3$ if $F(x^*) = 0$. In practice the procedure defined by (2.3.18), (2.3.19), (5.5.6) and (5.5.9) cannot be used as it stands because of the high probability of failure. It may, however, be incorporated into Algorithm 5.3.2 to give Algorithm 5.5.1 if Step 35 of Algorithm 5.3.2 is replaced with Step 35′, which is defined as follows.

Step 35′. If $-g^{(K)T} p^{(K)} \geqslant \varepsilon_5 \| g^{(K)} \|_2 \| p^{(K)} \|_2$ or $n_2 = 0$,

then go to Step 3.1, and the following steps are appended to Algorithm 5.3.2.

Step 3.1 If $n_5 = 0$ or $n_4 = 0$ or ($n_6 = 1$ and $n_7 = 0$) or

$g^{(K)T} p^{(K)} > 0$ or $\| p^{(K)} \|_2 \geqslant \varepsilon_6$ then go to Step 38.

Step 3.2 Compute $\bar{x}^{(K)} = x^{(K)} + p^{(K)}$

Step 3.3 Compute $\bar{A}^{(K)}$, $\bar{f}^{(K)}$, $\bar{B}^{(K)}$, $\bar{C}^{(K)}$, and $\bar{g}^{(K)}$ from

$$\bar{a}_l^{(K)} = a_l^{(K)} + G_l^{(K)} p^{(K)} \quad (l = 1, 2, \ldots, m)$$

$$\bar{A}^{(K)T} = \left[ \bar{a}_1^{(K)} \mid \cdots \mid \bar{a}_m^{(K)} \right]$$

$$\bar{f}_l^{(K)} = f_l^{(K)} + a_l^{(K)T} p^{(K)} + \tfrac{1}{2} p^{(K)T} G_l^{(K)} p^{(K)} \quad (1 \leqslant l \leqslant m),$$

$$\bar{B}^{(K)} = \sum_{l=1}^{m} \bar{f}_l^{(K)} \bar{G}_l^{(K)},$$

$$\bar{C}^{(K)} = \bar{A}^{(K)T} \bar{A}^{(K)} + \bar{B}^{(K)},$$

$$\bar{g}^{(K)} = \bar{A}^{(K)T} \bar{f}^{(K)}.$$

Step 3.4 Determine the modified Cholesky factors of $\bar{C}^{(K)}$ according to

$$L^{(K)} D^{(K)} L^{(K)T} = \bar{C}^{(K)} + E^{(K)}.$$

Step 3.5 Determine $\bar{p}^{(k)}$ from

$$L^{(k)} D^{(k)} L^{(k)T} \bar{p}^{(k)} = -\bar{g}^{(k)}.$$

Step 3.6 If $g^{(k)T} \bar{p}^{(k)} \geq 0$ then go to Step 38.

Step 3.7 Set $p^{(k)} = p^{(k)} + \bar{p}^{(k)}$, and go to Step 38. □

Clearly the sequence $(x^{(k)})$ generated from Algorithm 5.5.1 with Step 15 omitted converges to a critical point of F under the hypotheses of Theorem 2.3.4.

## 5.6 The Third Modification of Algorithm 5.3.2

As explained in section 5.1, if analytical formulae for $\partial_i \partial_j f_l(x)$ are not available, then $V_2^{(k)T} B^{(k)}$ may be estimated by using the finite-difference formula (5.1.29), or $B^{(k)}$ may be estimated by using a quasi-Newton updating formula such as (5.1.30). Gill and Murray (1976) observe that the quasi-Newton algorithm and the finite-difference algorithm require similar numbers of function and Jacobian evaluations but that the finite-difference algorithm will occasionally succeed where the quasi-Newton algorithm fails. Gill and Murray (1976) also observe that with large-residual problems a linear rate of convergence only is achieved near the solution when using quasi-Newton methods, in contrast with the superlinear rate which is often obtained for quasi-Newton methods for general unconstrained minimization. It would therefore appear that, if analytical formulae for $\partial_i \partial_j f_l(x)$ are not available, then the finite-difference method corresponding to (5.1.29) should be used

in preference to the quasi-Newton method corresponding to (5.1.30) - (5.1.33).

As shown in Table 6.4.4 the number of evaluations of A can be quite high compared with the number of evaluations of f, in spite of the fact that only n - r evaluations of A are required each time (5.1.29) is used. Therefore it is desirable to construct a method for estimating B which requires fewer evaluations of A than does the finite-difference method, and which is at least as efficient in other respects.

$$\text{If } A^{(k)T} = \left[\begin{array}{c:c:c:c} a_1^{(k)} & \cdots & \cdots & a_m^{(k)} \end{array}\right] \quad (k \geqslant 0) \tag{5.6.1}$$

where $a_i^{(k)}$ is column i of the n x m matrix $A^{(k)T}$, then for $k > 0$ an estimate $\bar{G}_i^{(k)}$ of $G_i^{(k)}$ is given by

$$\bar{G}_i^{(k)} = \frac{(a_i^{(k)} - a_i^{(k-1)})(x^{(k)} - x^{(k-1)})^T}{(x^{(k)} - x^{(k-1)})^T (x^{(k)} - x^{(k-1)})} \quad (1 \leqslant i \leqslant m). \tag{5.6.2}$$

This is a 'good' estimate of $G_i^{(k)}$ along the direction $x^{(k)} - x^{(k-1)}$ in that if $f \in C^2$ then

$$\begin{aligned} \| (\bar{G}_i^{(k)} - G_i^{(k)})(x^{(k)} - x^{(k-1)}) \| &= \| a_i^{(k)} - a_i^{(k-1)} - G_i^{(k)}(x^{(k)} - x^{(k-1)}) \| \\ &\leqslant \sup_{0 \leqslant t \leqslant 1} \| G_i(tx^{(k)} + (1-t)x^{(k-1)}) - G_i^{(k)} \| \\ &\qquad \| x^{(k)} - x^{(k-1)} \| , \end{aligned} \tag{5.6.3}$$

so if the sequence $(x^{(K)})$ converges then

$$\lim_{K \to \infty} \frac{\| (\bar{G}_i^{(K)} - G_i^{(K)}) (x^{(K)} - x^{(K-1)}) \|}{\| x^{(K)} - x^{(K-1)} \|} = 0 . \qquad (5.6.4)$$

If $k = 0$ then (5.6.2) is not applicable. Therefore if, in Algorithm 5.3.2, the Gauss-Newton step, which is always attempted initially when $k = 0$, is not adequate, then an alternative to (5.6.2) is required. If the singular value decomposition of $A^{(0)}$ is given by

$$A^{(0)} = U^{(0)} \begin{bmatrix} S^{(0)} \\ \hline 0 \end{bmatrix} V^{(0)T} ,$$

where $S^{(0)} = \text{Diag}\,(s_1^{(0)}, \ldots, s_n^{(0)})$, then the diagonal elements of $A^{(0)T} A^{(0)}$ are bounded according to

$$(A^{(0)T} A^{(0)})_{ii} \leq \max_{1 \leq j \leq n} \{ s_j^{(0)2} \} \quad (i = 1, 1, \ldots, n). \qquad (5.6.5)$$

Therefore a down-hill direction from $x^{(0)}$ may be determined by using Algorithm 5.3.2 with

$$B^{(0)} = (\max_{1 \leq i \leq n} \{ s_i^{(0)2} \}) I_n , \qquad (5.6.6)$$

where $I_n$ is the $n \times n$ unit matrix, when for $k = 0$, the Gauss-Newton direction from $x^{(0)}$ is inadequate.

The preceding ideas may be incorporated into Algorithm 5.3.2 to

give Algorithm 5.6.1 by replacing Step 23 with Step 23′ and Step 23″ which are defined as follows.

Step 23′ . If ($n_9 = 1$ or $n_4 = 1$) and $k \neq 0$ then Compute $B^{(k)}$ from

$$B^{(k)} = \sum_{i=1}^{m} f_i^{(k)} (a_i^{(k)} - a_i^{(k-1)}) (x^{(k)} - x^{(k-1)})^T / \| x^{(k)} - x^{(k-1)} \|_2^2$$

Step 23″ . If ($n_9 = 1$ or $n_4 = 1$) and $k = 0$ then compute $B^{(k)}$ from (5.6.6). □

The following theorem contains sufficient conditions for the sequence $(x^{(k)})$ generated from Algorithm 5.6.1 to converge to a unique solution $x^*$ of $g(x) = 0$, and for the rate of convergence to be at least linear.

Theorem 5.6.1

If 1. $f : \Omega \subset R^n \longrightarrow R^m$ is twice continuously differentiable and

$x^{(0)} \in R^n$ and $r > 0$ are such that $S \subset \Omega$, where $S = B(x^{(0)}, r)$;

2. for all $x \in S$ and for $i = 1, 2, \ldots, m$,

$$\| f_i(x) \| \leq \alpha_i ,$$

$$\| f_i'(x) \| \leq \beta_i ,$$

and

$$\| f_i''(x) \| \leq \gamma_i ;$$

3. $\| N'(x) \| \leq \Lambda \quad (\forall x \in S)$, where

$$N(x) = A(x)^{T} A(x) \ , \quad \text{in which}$$

$$A(x) = \left( \partial_j f_i(x) \right)_{m \times n} \ ;$$

4. $\| N(x^{(0)})^{-1} \| \leq M \ ;$

5. $M \Gamma < 1$, where

$$\Gamma = \Lambda r + \sum_{i=1}^{m} \alpha_i \gamma_i \ ;$$

6. $\exists \ \eta > 0$ such that

$$\| g(x^{(0)}) \| \leq \eta \ , \quad \text{where } g : \Omega \to R^n \text{ is defined}$$

by

$$g(x) = A(x)^{T} f(x) \ ;$$

7. $\exists \beta > 0$ such that

$$\| g(x) - g(y) - G(x^{(0)})(x - y) \| \leq \beta \| x - y \| \quad (\forall x, y \in S) \ ,$$

where $G(x) = N(x) + \sum_{i=1}^{m} f_i(x) G_i(x) \ ,$

in which

$$G_l(x) = \left( \partial_i \partial_j f_l(x) \right) \quad (l = 1, 2, \ldots, m) \ ;$$

8. $\tau = \lambda(\beta + \varepsilon) < 1$ , where

$$\varepsilon = \Lambda r + 2 \sum_{i=1}^{m} \alpha_i \gamma_i \ , \text{ and}$$

$$\lambda = M / (1 - M\Gamma) ;$$

9. $$\bar{r} = M\gamma / (1 - \tau)$$

$$\leqslant r,$$

then the sequence $(x^{(k)})$ generated from

$$x^{(k+1)} = x^{(k)} - T^{(k)-1} g(x^{(k)}) \quad (k \geqslant 0), \qquad (5.6.8)$$

where $$T^{(0)} = N(x^{(0)}) , \qquad (5.6.9)$$

and $$T^{(k)} = T(x^{(k)}, x^{(k-1)}) \quad (\forall k \geqslant 1) , \qquad (5.6.10)$$

in which

$$T(x, y) = N(x) + \sum_{i=1}^{m} f_i(x) (a_i(x) - a_i(y)) (x - y)^T / \|x - y\|_2^2 ,$$

$$(\forall x, y \in \Omega \text{ and } x \neq y) \qquad (5.6.11)$$

where $a_i(x) = \nabla f_i(x)$ , is well-defined and converges to the unique solution $x^*$ of $g(x) = 0$ in $B(x^{(0)}, \bar{r})$. Furthermore,

$$\|x^{(k)} - x^*\| \leqslant \tau^k \bar{r} \quad (\forall k \geqslant 0) .$$

Proof.

By Hypothesis 2, for i = 1, 2, ..., n we have

$$\| a_i(x) - a_i(y) \| \leq \gamma_i \| x - y \| \quad (\forall x, y \in S) \qquad (5.6.12)$$

and

$$\| N(x) - N(x^{(0)}) \| \leq \Lambda \| x - x^{(0)} \| \quad (\forall x \in S), \qquad (5.6.13)$$

and therefore by (5.6.11) and Hypothesis 5,

$$\| (T(x, y) - N(x^{(0)})) h \| / \| h \| \leq \Gamma \quad (\forall x, y \in S, x \neq y \text{ and } \forall h \in R^n). \qquad (5.6.14)$$

Therefore, by hypotheses 4, 5, and Banach's lemma, $T(x, y)^{-1}$ exists $(\forall x, y \in S, x \neq y)$, and

$$\| T(x, y)^{-1} \| \leq M / (1 - M\Gamma) = \lambda, \qquad (5.6.15)$$

where $\lambda$ is as in Hypothesis 8.

Now by (5.6.11), and hyptheses 7, 2, and 8 we have

$$\| (T(x, y) - G(x^{(0)})) h \| / \| h \| \leq \varepsilon \quad (\forall x, y \in S, x \neq y, h \in R^n).$$

Therefore,

$$\| T(x, y) - G(x^{(0)}) \| \leq \varepsilon \quad (\forall x, y \in S, x \neq y). \qquad (5.6.16)$$

Now, by (5.6.8) and hypotheses 6, 3, 8 and 9 we have

$$\| x^{(1)} - x^{(0)} \| \leqslant \| N (x^{(0)})^{-1} \| \cdot \| g (x^{(0)}) \|$$

$$\leqslant r, \qquad (5.6.17)$$

so that $x^{(1)} \in S$.

Also, by hypotheses 7, and 2 and (5.6.8) and (5.6.9) we have

$$\| g (x^{(1)}) \| \leqslant \| g (x^{(1)}) - g (x^{(0)}) - g'(x^{(0)})(x^{(1)} - x^{(0)}) \| + \| (g'(x^{(0)}) - T^{(0)})(x^{(1)} - x^{(0)}) \|$$

$$\leqslant \beta \| x^{(1)} - x^{(0)} \| + \| G (x^{(0)}) - N (x^{(0)}) \| \| x^{(1)} - x^{(0)} \|$$

and
$$\| G (x^{(0)}) - N (x^{(0)}) \| \leqslant \sum_{i=1}^{m} | f_i (x^{(0)}) | \| f_i'' (x^{(0)}) \|$$

$$\leqslant \varepsilon \quad ,$$

where $\varepsilon$ is defined in Hypothesis 8.

Therefore $\| g (x^{(1)}) \| \leqslant ( \beta + \varepsilon ) \| x^{(1)} - x^{(0)} \|$ . (5.6.18)

Now for k = 1 in (5.6.8) we have

$$x^{(2)} = x^{(1)} - T^{(1)-1} g (x^{(1)}) ,$$

where $T^{(1)} = T (x^{(1)}, x^{(0)})$, and $T^{(1)-1}$ exists

because $x^{(0)}$ , $x^{(1)} \in S$, and by (5.6.15)

$$\| T^{(1)-1} \| \leqslant \lambda$$

So $x^{(2)}$ is well-defined and by (5.6.18),

$$\| x^{(2)} - x^{(1)} \| \leqslant \lambda (\beta + \varepsilon) \| x^{(1)} - x^{(0)} \| \qquad (5.6.19)$$

By (5.6.19) and Hypothesis 8, $x^{(2)} \in S$.

Suppose that for some $l \geqslant 1$, we have for $k = 1,2,\ldots, l$,

$$\| x^{(k)} - x^{(0)} \| \leqslant r, \qquad (5.6.20)$$

$$\| x^{(k)} - x^{(k-1)} \| \leqslant \lambda \| g(x^{(k-1)}) \|, \qquad (5.6.21)$$

$$\| g(x^{(k)}) \| \leqslant (\beta + \varepsilon) \| x^{(k)} - x^{(k-1)} \| , \qquad (5.6.22)$$

which by (5.6.17 ) and (5.6.18) are true for k = 1.

Then

$$T^{(l)} = T(x^{(l)} , x^{(l-1)}),$$

whence because $x^{(l)}$, $x^{(l-1)} \in S$, it follows that $T^{(l)-1}$ exists,

and $\| T^{(l)-1} \| \leqslant \lambda$. So by (5.6.8)

$$\| x^{(l+1)} - x^{(l)} \| \leqslant \lambda \| g(x^{(l)}) \| , \qquad (5.6.23)$$

whence (5.6.21) holds for k = $l + 1$. Therefore, from (5.6.23) and (5.6.22),

$$\| x^{(l+1)} - x^{(l)} \| \leq \lambda ( \beta + \varepsilon ) \| x^{(l)} - x^{(l-1)} \|$$

$$= \tau \| x^{(l)} - x^{(l-1)} \|.$$

Also by (5.6.21), (5.6.22) for k = 1, 2, ..., $(l - 1)$

$$\| x^{(k+1)} - x^{(k)} \| \leq \tau \| x^{(k)} - x^{(k-1)} \| .$$

So

$$\| x^{(k+1)} - x^{(k)} \| \leq \tau \| x^{(k)} - x^{(k-1)} \| \quad (k = 1, 2, \ldots, l ).$$

So

$$\| x^{(k+1)} - x^{(k)} \| \leq \tau^{k} \| x^{(1)} - x^{(0)} \| (k = 0, 1, \ldots, l ). \tag{5.6.24}$$

So

$$\| x^{(l+1)} - x^{(0)} \| \leq \sum_{k=0}^{l} \| x^{(k+1)} - x^{(k)} \|$$

$$\leq \| x^{(1)} - x^{(0)} \| \sum_{k=0}^{l} \tau^{k}$$

$$\leq M \eta / (1 - \tau).$$

Thus, by Hypothesis 9, $x^{(l+1)} \in S$, and (5.6.20) holds for k = $l + 1$.

By (5.6.8), (5.6.16) and Hypothesis 7, it follows that

$$\| g (x^{(l+1)}) \| \leq \| g (x^{(l+1)}) - g (x^{(l)}) - g'(x^{(0)}) (x^{(l+1)} - x^{(l)}) \|$$

$$+ \| (g'(x^{(0)}) - T^{(l)}) (x^{(l+1)} - x^{(l)}) \|$$

$$\leqslant \beta \| x^{(l+1)} - x^{(l)} \| + \| g'(x^{(0)}) - T^{(l)} \| \| x^{(l+1)} - x^{(l)} \|$$

$$\leqslant ( \beta + \varepsilon ) \| x^{(l+1)} - x^{(l)} \| ,$$

so that (5.6.22) holds for $k = l + 1$. Therefore, by induction (5.6.20)- (5.6.22) hold ($\forall\, k \geqslant 1$). So by (5.6.20), $x^{(k)} \in S$ ($\forall\, k \geqslant 0$).

Now for any $k > 0$, by (5.6.24),

$$\| x^{(l+k)} - x^{(l)} \| \leqslant \sum_{i=1}^{k} \| x^{(l+i)} - x^{(l+i-1)} \|$$

$$\leqslant M \eta \tau^{l} / (1 - \tau).$$

Thus, by Hypothesis 9,

$$\| x^{(l+k)} - x^{(l)} \| \leqslant \tau^{l-1} \bar{r} \quad (\forall\, l , k \geqslant 0).$$

So because $\tau < 1$, $(x^{(k)})$ is a Cauchy sequence which has a limit $x^{*} \in R^{n}$, with

$$\| x^{(k)} - x^{*} \| \leqslant \tau^{k} \bar{r} \quad (\forall k \geqslant 0) . \qquad (5.6.25).$$

Therefore by continuity of $g$

$$g (x^{*}) = 0 .$$

If $x^{**}$ is any other solution of $g(x) = 0$, such that $x^{**} \in S$,

then

$$\|x^{**} - x^{*}\| = \|T^{(0)-1}\, T^{(0)}(x^{**} - x^{*})\|$$

$$\leqslant \|T^{(0)-1}\| \|g(x^{**}) - g(x^{*}) - T^{(0)}(x^{**} - x^{*})\|$$

$$\leqslant \lambda(\beta + \varepsilon)\, \|x^{**} - x^{*}\|$$

$$= \tau \|x^{**} - x^{*}\|$$

$$< \|x^{**} - x^{*}\| \quad .$$

This is impossible, so $x^{*} \in S$ is unique. □

Lemma 5.6.1

If 1. the hypotheses of Theorem 5.6.1 hold;

2. $\exists \alpha > 0$ and $\exists \beta > 1$ such that

$$\| a_i(x) - a_i(y) \| \leqslant \alpha \| x - y \|^{\beta} \quad ;$$

3. the sequence $(x^{(k)})$ is generated from (5.6.8) - (5.6.10),

then $T^{(k)} \longrightarrow N(x^{*})$ (as $k \to \infty$).

Proof.

From (5.6.11) and (5.6.10) we have

$$\| T^{(k)} - N(x^{*}) \| \leqslant \| N(x^{(k)}) - N(x^{*}) \|$$

$$+ \sum_{i=1}^{m} |f_i(x^{(k)})| \| (a_i(x^{(k)}) - a_i(x^{(k-1)}))(x^{(k)} - x^{(k-1)})^{T} \| / \| x^{(k)} - x^{(k-1)} \|$$

Therefore, by hypotheses 3 and 2 we will have

$$\| T^{(k)} - N(x^*) \| \longrightarrow 0 \quad (\text{as } k \longrightarrow \infty). \quad \square$$

Theorem 5.6.2

If the hypotheses 1 - 3 of Lemma 5.6.1 hold then the convergence of $(x^{(k)})$ is Q-superlinear.

Proof.

By Theorem 1.2.10, the truth of Theorem 5.6.2 follows if we prove that

$$\| (T^{(k)} - G(x^*))(x^{(k)} - x^{(k-1)}) \| \; / \; \| x^{(k)} - x^{(k-1)} \| \longrightarrow 0 \quad \text{as } k \longrightarrow \infty.$$

By Lemma 5.6.1 we have

$$T^{(k)} - T^{(k-1)} \longrightarrow 0 \quad (\text{as } k \longrightarrow \infty).$$

Now

$$\| (T^{(k)} - G(x^*)) s^{(k-1)} \|_2 \; / \; \| s^{(k-1)} \|_2$$

$$\leqslant \| (T^{(k)} - G^{(k)}) s^{(k-1)} \|_2 \; / \; \| s^{(k-1)} \|_2 +$$

$$\| (G^{(k)} - G(x^*)) s^{(k-1)} \|_2 \; / \; \| s^{(k-1)} \|_2 , \qquad (5.6.27)$$

where $s^{(k-1)} = x^{(k)} - x^{(k-1)}$ and $G^{(k)} = G(x^{(k)})$ is

defined as in Hypothesis 7 of Theorem 5.6.1.

But

$$\| (G^{(K)} - G(x^*)) s^{(K-1)} \|_2 / \| s^{(K-1)} \|_2 \leq \| G^{(K)} - G(x^*) \|_2$$

and therefore the second term of the right hand side of (5.6.27) tends to zero as $k \to \infty$. Also, by (5.6.11) we have

$$\| (T^{(K)} - G^{(K)}) s^{(K-1)} \|_2 / \| s^{(K-1)} \|_2$$

$$\leq \sum_{l=1}^{m} q^{(K)} | f_l(x^{(K)}) | \| a_l(x^{(K)}) - a_l(x^{(K-1)}) - G_l(x^{(K-1)}) s^{(K-1)} \|_2$$

$$\leq \sum_{l=1}^{m} | f_l(x^{(K)}) | \| G_l(\xi^{(K)}) - G_l(x^{(K-1)}) \|_2 , \qquad (5.6.28)$$

where $q^{(K)} = 1 / \| s^{(K-1)} \|_2$

When $k \to \infty$, the right hand side of (5.6.28) will tend to zero, and thus the theorem holds. □

It seems that Hypothesis 2 of Lemma 5.6.1 is stronger than it need be, since, as Table 6.4.4 shows, the sequence $(x^{(K)})$ generated from Algorithm 5.6.1 appears to be Q-superlinearly convergent for problems 2.13 and 2.16, while none of these functions satisfies the condition of Hypothesis 2 of Lemma 5.6.1.

# Chapter 6.

## Computational Results and Conclusions.

### 6.1 Introduction

The algorithms which are presented in chapters 3 - 5 have been used to solve a group of test problems which are listed in appendices 1 and 2. The algorithms have been implemented on the IBM 360/44 Computer at St. Andrews with double precision arithmetic to obtain the computational results which are presented in this chapter.

In this chapter $t_c$ is the CPU time, in seconds, rounded to the nearest 0.02 sec., required for the execution of a given algorithm, and is independent of operator intervention time and of the time required by other tasks which may be simultaneously executed by the computer.

### 6.2 Comparison of the Algorithms from Chapter 3.

In order to compare the efficiencies of the algorithms which have been described in Chapter 3, they have been used to solve all of the test problems which are listed in Appendix 1.

Algorithms 3.2.5 - 3.2.7 have also been compared with Algorithm 1.5.3. Furthermore, the Cholesky factorization, steplength algorith (Algorithm 1.2.2), and convergence criteria sub-programs which are used in the implementations of all algorithms to obtain the numerical results in sections 6.2 and 6.3 are identical with those which are used in the Numerical Algorithms Group (NAG) implementation of Algorithm 3.1.1, and it is reasonable to suppose that the differences in the computational results which are obtained are due solely to the algorithmic differences, and not to variations in programming

efficiency.

The values of the parameters which have been used with the algorithms in this section are $\epsilon_1 = 2^{-28}$, $\epsilon_2 = 10^{-10}$, $\epsilon_3 = 20$, $\epsilon_4 = 10^{-1}$, $\eta = 0.9$, $\mu = 10^{-4}$, and $\lambda = 1$, where $\lambda$ is a bound for the steplength, and this bound has been recommended in the NAG Manual by Gill and Murray. Also, $p = 3$ gives almost optimal efficiency for all of the test problems of Appendix 1. In each case, it is supposed that analytical formulae for F and for the n components of the gradient g of F are known. It is often true that the formulae for the components of g contain expressions in common, so that the computational labour which is required in order to evaluate g is often less than n times that which is required for the evaluation of one component of g. Furthermore, it is often true that the formula for F contains expressions which are common to some or all of the formulae for the components of g. In this case considerable saving in computational labour can be made in the simultaneous evaluation of F and g.

It is difficult, for the reasons which have been given in the preceding paragraph, to allocate an appropriate weighting factor to an evaluation of g relative to an evaluation of F when attempting to compare the computational labour required to evaluate g with that required to evaluate F. If such a factor could be found then it could be used to estimate the relative computational efficiencies of the algorithms for the numerical estimation of the Hessian G of F which are described in Chapter 4.

Let the number of arithmetical operations required to compute F and g be $m_F$ and $m_g$ respectively. Then

$$m_g = m_F + n\nu , \qquad (6.2.1)$$

where $\nu$ is the average number of arithmetical operations which are required to evaluate each component of $g$, in addition to $m_F$. According to Fletcher (1972) a typical value for $n\nu / m_F$ is about 2. Therefore, if we regard one evaluation of $F$ as requiring one unit of computational labour, the number of units of computational labour required for one evaluation of $g$ is about 3.

If in implementation of any algorithm discussed in chapters 3 and 4, the number of evaluations of $F$ is $n_F$, the number of evaluations of $g$, excluding those which are required to estimate $G$, is $n_g$, and the number of evaluations of $G$ is $n_G$, then an index $n_c$ of computational labour for algorithms 3.1.2, 3.2.2, 3.2.4, and 3.2.5 - 3.2.7, excluding all arithmetical operations save those which are required to evaluate $F$, $g$, and $G$, is given by

$$n_c = n_F + 3n_g + l_G n_G \qquad (6.2.2)$$

in which $l_G$ is the number of units of computational labour which are required for one evaluation of $G$.

In general, the value of $l_G$ depends upon the value of $\mu = n\nu / m_F$. In practice, the value of $\mu$ varies widely. In some cases $\mu \approx 0$, while in others $\mu \approx n - 1$. Therefore the index of computational labour given by (6.2.2) is not always realistic. Consequently it is necessary to use an additional index of computational labour, namely the CPU time $t_c$, which is defined in Section 6.1.

Every attempt has been made, when obtaining the numerical results which are presented in this section, to economize on computational

labour when computing g. Thus, any expression which is common to some or all of the components of g is evaluated before evaluating g and is used as many times as required in the computation of g itself; this results in $\mu = 2$ being rather an over-estimate for the set of test problems which are used in this section.

Moreover, in algorithms 3.1.2, 3.2.2, and 3.2.4 - 3.2.7, g must be calculated at several points distinct from those at which F and G are calculated. Therefore no allowance has been made in (6.2.2) for the fact that at least once in each iteration, F and g are evaluated at the same point. Therefore we set $l_G = 3n$ for the algorithms 3.1.2, 3.2.2 and 3.2.4 - 3.2.7, while $l_G = 0$ for Algorithm 1.5.3. Algorithms 3.1.2, 3.2.2, 3.2.4 - 3.2.7 and 1.5.3 have been used to solve all of the test problems listed in Appendix 1.

Tables 6.2.1, 6.2.2, 6.2.3 and 6.2.4 show the values of $n_F$, $n_g$, $n_G$, $n_c$ and $t_c$ obtained by solving all of the test problems in Appendix 1 which do not have a sparse Hessian by using the algorithms 3.1.2, 3.2.2, 3.2.4 - 3.2.7 and 1.5.3. Table 6.2.5 (a) contains the total number $N_c$ of units of computational labour and the total computing time $T_c$ over all of the test problems. Table 6.2.6 contains the iteration numbers which each algorithm requires to solve each of the test problems.

If the algorithms are compared in terms of $n_c$ then from tables 6.2.1 - 6.2.4 the following conclusions may be drawn.

1. Algorithm 3.2.2 is superior to Algorithm 3.1.2 for all problems save problems 1.1 and 1.21.
2. Algorithm 3.2.2 is superior to Algorithm 3.2.4 for all problems save problems 1.1, 1.5, 1.21 and 1.22.

3. Algorithm 3.2.4 is superior to Algorithm 3.1.2 for all problems save problems 1.1, 1.2, 1.8 and 1.9.
4. Algorithm 3.2.5 is superior to algorithms 3.2.2 and 3.2.4 for all problems save problems 1.21 and 1.22; Algorithm 3.2.5 is also superior to Algorithm 3.1.2 for all problems save problem 1.21 and 1.22.
5. Algorithm 3.2.6 is superior to Algorithm 3.2.5 for all problems save problem 1.17.
6. Algorithm 3.2.6 is superior to Algorithm 3.2.4 for all problems save problems 1.21 and 1.22; it is also superior to Algorithm 3.2.2 for all problems save problems 1.17, 1.21 and 1.22. Also Algorithm 3.2.6 is superior to Algorithm 3.1.2 for all problems save problem 1.21 and 1.22.
7. Algorithm 1.5.3 is superior to algorithms 3.1.2 and 3.2.4 for all problems, but it is inferior to Algorithm 3.2.2 for Problem 1.9.
8. Algorithm 1.5.3 is superior to Algorithm 3.2.5 for all problems save problems 1.2, 1.6, 1.8, 1.9, 1.17 and 1.20.
9. Algorithm 1.5.3 is superior to Algorithm 3.2.6 for all problems save problems 1.2, 1.6, 1.8, 1.9, 1.19 and 1.20.
10. Algorithm 3.2.7 is inferior to Algorithm 1.5.3 for all problems save problems 1.6, 1.9, 1.20 and algorithms 3.2.7 and 1.5.3 are equally efficient for Problem 1.8.
11. Algorithm 3.2.7 is superior to Algorithm 3.2.6 for problems 1.21and 1.17; Algorithm 3.2.7 is superior to Algorithm 3.2.5 for problems 1.1, 1.9, 1.21 and 1.22.
12. Algorithm 3.2.7 is superior to Algorithm 3.2.4 for all problems save problems 1.21 and 1.22. Also, Algorithm 3.2.7

is superior to Algorithms 3.1.2 and 3.2.2 for all problems save problems 1.21 and 1.22.

If we were to solve those test problems of Appendix 1 which do not have sparse Hessians, then from the values of $N_c$ in Table 6.2.5 (a), it can be concluded that Algorithm 1.5.3 is the best algorithm and the second best is Algorithm 3.2.2, for functions which are expensive to compute.

If we compare the algorithms in terms of CPU time $t_c$, then from tables 6.2.1 - 6.2.4 the following conclusions may be drawn.

1. Algorithm 3.2.2 is superior to algorithms 3.1.2 and 3.2.4 for all problems save problems 1.5, 1.21 and 1.2 2. Also, Algorithm 3.2.2 is as efficient as Algorithm 3.1.2 for Problem 1.1.
2. Algorithm 3.2.2 is superior to Algorithm 3.2.5 for all problems save problems 1.6, 1.16, 1.17 and 1.18. Algorithm 3.2.2 is also superior to Algorithm 3.2.6 for all problems save problems 1.6, 1.8, 1.16 and 1.18.
3. Algorithm 3.2.7 is superior to Algorithm 3.2.2 for problems 1.6, 1.16, and 1.18.
4. Algorithm 1.5.3 is superior to Algorith 3.2.5 for problems 1.16, 1.18, 1.21 and 1.22. Also, Algorithm 1.5.3 is superior to Algorithm 3.2.6 for problems 1.16, 1.17, 1.18, 1.21 and 1.22.
5. Algorithm 1.5.3 is superior to Algorithm 3.2.7 for problems 1.16, 1.17, 1.18, 1.21 and 1.22.
6. Algorithm 3.2.5 is superior to Algorithm 3.2.6 for all problems save problems 1.7, 1.8, 1.9, 1.16, 1.21, and 1.22 and

algorithms 3.2.5 and 3.2.6 are equally efficient for problems 1.1, 1.2.

7. Algorithm 3.2.7 is superior to Algorithm 3.2.5 for problems 1.16, 1.21 and 1.22. Also, Algorithm 3.2.7 is superior to Algorithm 3.2.6 for problems 1.17 and 1.21.

If we were to solve problems 1.1, 1.2, 1.4 - 1.9 and 1.16 - 1.22 by algorithms 3.1.2, 3.2.2, 3.2.4 - 3.2.7 and 1.5.3, then from Table 6.2.5 (a) it may be concluded that Algorithm 3.2.5 is the best algorithm and Algorithm 1.5.3 is superior to all algorithms save Algorithm 3.2.5. Also, Algorithm 3.2.2 is significantly more efficient than algorithms 3.1.2 and 3.2.4.

As is shown in tables 6.2.1 - 6.2.4, algorithms 3.2.5 - 3.2.7 and 1.5.3 are inefficient for problems 1.21 and 1.22. In particular, algorithms 3.2.5 - 3.2.7 require more evaluations of the Hessian than if these problems were solved by using algorithms 3.2.2, 3.2.4 and 3.1.2 respectively. Thus, if we omit problems 1.21 and 1.22 from the comparison then from Table 6.2.5 (b) can be concluded that in terms of the total CPU time required to solve problems 1.1, 1.2, 1.4 - 1.9, and 1.16 - 1.20, Algorithm 3.2.5 is significantly more efficient than algorithms 3.1.2, 3.2.2, 3.2.4, 3.2.6, 3.2.7 and 1.5.3; while in terms of computational labour, Algorithm 1.5.3 is more efficient than the others. The inferiority of algorithms 3.2.6 and 3.2.7 in terms of $T_c$ is inherited from algorithms 3.1.2 and 3.2.4 which are inferior to Algorithm 3.2.2.

In Algorithm 3.2.5, (1.5.3) is used to estimate the Hessian until for some k, $\| g^{(k)} \| \leq \varepsilon_4$. Then $x^{(k)}$ is used as a starting point for the Newton-type algorithm 3.2.2.

One would expect that for small values of $\epsilon_4$, $x^{(k)}$ would be close to a minimizer $x^*$ and therefore would be a good starting point for Algorithm 3.2.2.

For problems 1.1, 1.2, 1.4 - 1.9, and 1.16 - 1.20 computational experience shows this conjecture to be valid. For problems 1.21, and 1.22, however, it is found that with $\epsilon_4 = 10^{-1}$, $x^{(k)}$ is a poor starting point for Algorithm 3.2.2.

For $\epsilon_4 = 10^{-5}$, $x^{(k)}$ is found to be a satisfactory starting point for Algorithm 3.2.2 when applied to problems 1.21 and 1.22, but for Problem 1.22, the additional number of iterations corresponding to the use of updating formula (1.5.3) for estimating the Hessian give rise to large overall values of $t_c$. See Table 6.2.5 (c).

For problems 1.1, 1.2, 1.4 - 1.9 and 1.16 - 1.22 computational experience[1] shows that the value of $T_c$ corresponding to the Newton-type algorithm 3.1.2 of Gill and Murray is 717 Sec., while for the quasi-Newton algorithm 1.5.3 $T_c$ is 318 Sec.. For Algorithm 3.2.5, $T_c$ is 178 Sec.. It would therefore appear that Algorithm 3.2.5 is significantly better than Algorithm 3.2.2 and is also better than Algorithm 1.5.3 at least for the test problems which were used.

Algorithms 3.1.1, 3.2.1, 3.2.3 and 3.3.1 - 3.3.6 have been used to solve the test problems of Appendix 1, using analytical formulae for the Hessians. It has been observed that, $p = 2$ gives almost optimal efficiency for algorithms 3.2.1, 3.2.3 and 3.3.1 - 3.3.6, which suggests that higher order methods are not applied often. This is verified from a detailed print-out of the computational results. This is because either the search directions after an extension were not down-hill or because the sufficient decrease

---

(1) see Table 6.2.5(a)

condition required for allowing the extensions is not satisfied. However, among algorithms 3.2.1, 3.2.3 and 3.3.1 - 3.3.6, algorithms 3.2.1, 3.3.1, 3.3.3, and 3.3.5 are the best on problems 1.1 - 1.10, when p = 2. The tables 6.2.7 and 6.2.8 show the CPU times $t_c$ which have been obtained in solving problems 1.1 - 1.10. From Table 6.2.7 the following conclusions may be drawn.

1. Algorithm 3.2.1 is superior to Algorithm 3.1.1 for all ten test problems, and is as efficient as Algorithm 3.3.3 for problems 1.1 and 1.8.
2. Algorithm 3.3.3 is superior to Algorithm 3.1.1 for all ten problems save problems 1.6 and 1.9.
3. Algorithm 3.3.3 is inferior to Algorithm 3.2.1 for problems 1.6, 1.9 and 1.10.
4. Algorithm 3.3.5 is inferior to Algorithm 3.2.1 for all problems save problem 1.9
5. Algorithm 3.3.1 is superior to Algorithm 3.2.1 for seven problems and is equally efficient to Algorithm 3.2.1 for Problem 1.1.

From Table 6.2.8, which shows the total CPU time $T_c$ over the problems 1.1 - 1.10, it may be concluded that Algorithms 3.3.3 is superior to algorithms 3.1.1, and 3.3.5 and requires over 5 % less computing time than is required by Algorithm 3.3.1.

| PROBLEM | Algorithm 3.1.2 | | | | | Algorithm 3.2.2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
| 1.1 | 24 | 24 | 21 | 222 | 0.38 | 44 | 42 | 16 | 266 | 0.38 |
| 1.2 | 53 | 53 | 45 | 752 | 1.76 | 79 | 70 | 29 | 637 | 1.42 |
| 1.4 | 21 | 21 | 21 | 336 | 0.80 | 37 | 37 | 12 | 292 | 0.62 |
| 1.5 | 15 | 15 | 14 | 186 | 0.66 | 23 | 21 | 11 | 185 | 0.74 |
| 1.6 | 20 | 20 | 19 | 194 | 2.28 | 23 | 20 | 9 | 137 | 1.66 |
| 1.7 | 12 | 12 | 12 | 120 | 1.54 | 14 | 13 | 5 | 83 | 1.04 |
| 1.8 | 13 | 13 | 13 | 159 | 2.28 | 14 | 12 | 7 | 113 | 1.54 |
| 1.9 | 15 | 15 | 15 | 240 | 3.76 | 25 | 19 | 10 | 202 | 3.14 |
| 1.16 | 60 | 60 | 47 | 1086 | 28.94 | 79 | 71 | 34 | 904 | 24.90 |
| 1.17 | 28 | 28 | 28 | 868 | 564.22 | 23 | 23 | 8 | 308 | 202.74 |
| 1.18 | 13 | 13 | 13 | 276 | 75.82 | 23 | 23 | 8 | 236 | 60.36 |
| 1.19 | 55 | 55 | 48 | 1660 | 8.00 | 69 | 68 | 25 | 1023 | 5.14 |
| 1.20 | 30 | 30 | 30 | 1020 | 5.10 | 35 | 34 | 12 | 497 | 2.66 |
| 1.21 | 157 | 157 | 110 | 1948 | 17.08 | 275 | 241 | 94 | 2126 | 18.92 |
| 1.22 | 37 | 37 | 30 | 508 | 4.44 | 60 | 59 | 22 | 501 | 4.68 |

Table 6.2.1

| PROBLEM | Algorithm 3.2.4 | | | | | Algorithm 3.2.5 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
| 1.1 | 22 | 44 | 16 | 250 | 0.42 | 43 | 41 | 2 | 178 | 0.52 |
| 1.2 | 52 | 88 | 37 | 760 | 1.82 | 86 | 84 | 2 | 362 | 2.60 |
| 1.4 | 14 | 38 | 14 | 296 | 0.78 | 42 | 42 | 8 | 264 | 0.92 |
| 1.5 | 12 | 20 | 11 | 171 | 0.64 | 30 | 29 | 1 | 126 | 0.86 |
| 1.6 | 18 | 22 | 17 | 186 | 2.06 | 20 | 19 | 2 | 89 | 1.20 |
| 1.7 | 10 | 14 | 10 | 112 | 1.36 | 17 | 16 | 2 | 77 | 1.20 |
| 1.8 | 12 | 16 | 12 | 168 | 2.26 | 20 | 20 | 2 | 98 | 1.58 |
| 1.9 | 15 | 20 | 14 | 243 | 3.76 | 36 | 33 | 5 | 195 | 3.52 |
| 1.16 | 50 | 78 | 38 | 968 | 25.64 | 85 | 78 | 27 | 805 | 23.20 |
| 1.17 | 24 | 32 | 24 | 768 | 502.32 | 31 | 31 | 3 | 205 | 138.10 |
| 1.18 | 10 | 22 | 9 | 238 | 63.28 | 32 | 31 | 2 | 161 | 42.14 |
| 1.19 | 47 | 77 | 39 | 1448 | 7.44 | 80 | 79 | 18 | 857 | 8.00 |
| 1.20 | 27 | 43 | 26 | 936 | 4.78 | 43 | 42 | 6 | 349 | 5.54 |
| 1.21 | 143 | 243 | 83 | 1868 | 16.34 | 575 | 529 | 189 | 4430 | 39.66 |
| 1.22 | 30 | 58 | 21 | 456 | 4.12 | 132 | 123 | 43 | 1017 | 9.40 |

Table 6.2.2

| | Algorithm 3.2.6 | | | | | Algorithm 1.5.3 | | | |
|---|---|---|---|---|---|---|---|---|---|
| PROBLEM | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_F$ | $n_g$ | $n_c$ | $t_c$ |
| 1.1 | 35 | 40 | 1 | 161 | 0.52 | 39 | 39 | 156 | 0.74 |
| 1.2 | 81 | 83 | 1 | 352 | 2.60 | 93 | 93 | 372 | 3.04 |
| 1.4 | 25 | 41 | 8 | 244 | 1.10 | 50 | 50 | 200 | 1.96 |
| 1.5 | 27 | 29 | 1 | 123 | 0.94 | 29 | 29 | 116 | 1.46 |
| 1.6 | 15 | 19 | 2 | 84 | 1.22 | 26 | 26 | 104 | 2.34 |
| 1.7 | 14 | 16 | 2 | 74 | 1.06 | 18 | 18 | 72 | 1.68 |
| 1.8 | 15 | 19 | 2 | 90 | 1.52 | 26 | 26 | 104 | 2.30 |
| 1.9 | 28 | 32 | 5 | 184 | 3.40 | 59 | 58 | 232 | 5.06 |
| 1.16 | 62 | 78 | 27 | 782 | 22.38 | 128 | 128 | 512 | 17.58 |
| 1.17 | 33 | 34 | 9 | 378 | 244.08 | 71 | 71 | 284 | 174.82 |
| 1.18 | 27 | 31 | 2 | 156 | 42.28 | 35 | 35 | 140 | 38.94 |
| 1.19 | 57 | 79 | 18 | 834 | 8.16 | 233 | 233 | 932 | 23.32 |
| 1.20 | 35 | 43 | 6 | 344 | 5.62 | 108 | 108 | 432 | 12.04 |
| 1.21 | 397 | 619 | 181 | 4426 | 38.86 | 441 | 441 | 1764 | 27.12 |
| 1.22 | 77 | 127 | 39 | 926 | 8.46 | 95 | 95 | 380 | 6.08 |

Table 6.2.3

Algorithm 3.2.7

| PROBLEM | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
|---|---|---|---|---|---|
| 1.1 | 39 | 39 | 2 | 168 | 0.58 |
| 1.2 | 90 | 90 | 2 | 384 | 2.70 |
| 1.4 | 30 | 30 | 13 | 276 | 1.10 |
| 1.5 | 28 | 28 | 2 | 130 | 0.94 |
| 1.6 | 17 | 17 | 4 | 92 | 1.32 |
| 1.7 | 15 | 15 | 3 | 78 | 1.20 |
| 1.8 | 17 | 17 | 4 | 104 | 1.74 |
| 1.9 | 29 | 29 | 6 | 188 | 3.56 |
| 1.16 | 62 | 62 | 31 | 806 | 22.78 |
| 1.17 | 31 | 31 | 8 | 340 | 228.50 |
| 1.18 | 28 | 28 | 3 | 166 | 44.92 |
| 1.19 | 60 | 60 | 24 | 960 | 8.88 |
| 1.20 | 38 | 38 | 8 | 392 | 5.80 |
| 1.21 | 352 | 352 | 228 | 4144 | 36.46 |
| 1.22 | 87 | 87 | 52 | 972 | 8.90 |

Table 6.2.4

| Algorithm | 3.1.2 | 3.2.2 | 3.2.4 | 3.2.5 | 3.2.6 | 3.2.7 | 1.5.3 |
|---|---|---|---|---|---|---|---|
| $N_c$ | 9575 | 7510 | 8868 | 9413 | 9248 | 9200 | 5800 |
| $T_c$ | 717.06 | 329.94 | 637.02 | 278.44 | 382.20 | 369.38 | 318.48 |

(The values of $N_c$ and $T_c$ for those test problems in Appendix 1 for which the Hessian is not sparse.)

Table 6.2.5 (a)

| Algorithm | 3.1.2 | 3.2.2 | 3.2.4 | 3.2.5 | 3.2.6 | 3.2.7 | 1.5.3 |
|---|---|---|---|---|---|---|---|
| $N_c$ | 7119 | 4883 | 6544 | 3966 | 3896 | 4084 | 3676 |
| $T_c$ | 695.54 | 306.34 | 616.56 | 229.38 | 334.88 | 324.02 | 285.29 |

(The values of $N_c$ and $T_c$ for those test problems in Appendix 1 for which the Hessian is sparse, but excluding problems 1.21 and 1.22.)

Table 6.2.5 (b)

| PROBLEM | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_I$ | Iteration Number |
|---|---|---|---|---|---|---|---|
| 1.16 | 152 | 152 | 1 | 626 | 20.96 | 138 | |
| 1.19 | 160 | 160 | 2 | 700 | 20.10 | 130 | |
| 1.21 | 487 | 485 | 8 | 2038 | 27.50 | 359 | |
| 1.22 | 197 | 196 | 2 | 809 | 11.40 | 156 | |

Table 6.2.5 (c)

(Computational Results corresponding to $\varepsilon_4 = 10^{-5}$ in Algorithm 3.2.5)

Iteration Number $n_I$

| Algorithm | 3.1.2 | 3.2.2 | 3.2.4 | 3.2.5 | 3.2.6 | 1.5.3 |
|---|---|---|---|---|---|---|
| PROBLEM | | | | | | |
| 1.1 | 21 | 16 | 16 | 30 | 29 | 32 |
| 1.2 | 45 | 29 | 37 | 69 | 68 | 75 |
| 1.4 | 21 | 12 | 14 | 23 | 23 | 49 |
| 1.5 | 14 | 11 | 11 | 26 | 26 | 29 |
| 1.6 | 19 | 9 | 17 | 15 | 15 | 22 |
| 1.7 | 12 | 5 | 10 | 14 | 14 | 18 |
| 1.8 | 13 | 7 | 12 | 13 | 13 | 24 |
| 1.9 | 15 | 10 | 14 | 23 | 23 | 36 |
| 1.16 | 47 | 34 | 38 | 45 | 45 | 112 |
| 1.17 | 28 | 8 | 24 | 21 | 27 | 69 |
| 1.18 | 13 | 8 | 9 | 24 | 24 | 34 |
| 1.19 | 48 | 25 | 39 | 47 | 47 | 195 |
| 1.20 | 30 | 12 | 26 | 35 | 35 | 102 |
| 1.21 | 110 | 94 | 83 | 201 | 192 | 332 |
| 1.22 | 30 | 22 | 21 | 54 | 50 | 76 |

Table 6.2.6

| Algorithm | 3.1.1 | 3.2.1 | 3.3.1 | 3.3.3 | 3.3.5 |
|---|---|---|---|---|---|
| PROBLEM | | | | | |
| 1.1 | 0.34 | 0.30 | 0.30 | 0.30 | 0.34 |
| 1.2 | 1.28 | 1.08 | 1.06 | 1.04 | 1.28 |
| 1.3 | 0.84 | 0.76 | 0.70 | 0.62 | 0.84 |
| 1.4 | 0.60 | 0.50 | 0.48 | 0.42 | 0.60 |
| 1.5 | 0.54 | 0.50 | 0.48 | 0.44 | 0.60 |
| 1.6 | 1.94 | 1.34 | 1.30 | 1.96 | 1.40 |
| 1.7 | 1.28 | 0.88 | 0.82 | 0.86 | 1.00 |
| 1.8 | 1.76 | 1.18 | 1.16 | 1.18 | 1.48 |
| 1.9 | 2.98 | 2.66 | 3.26 | 3.60 | 2.16 |
| 1.10 | 2.48 | 2.00 | 2.24 | 2.02 | 2.64 |

Table 6.2.7 (values of $t_c$ sec.)

| Algorithm | 3.1.1 | 3.2.1 | 3.3.1 | 3.3.3 | 3.3.5 |
|---|---|---|---|---|---|
| $T_c$ Sec. | 14.04 | 11.20 | 11.80 | 11.00 | 13.98 |

Table 6.2.8

## 6.3 Comparison of the Methods for Estimating the Hessian Numerically.

The numerical methods for estimating the Hessian matrix which are described in Chapter 4 have been incorporated into Algorithm 3.1.2 to solve the test problems in Appendix 1, and the numerical results are shown in tables 6.3.1 - 6.3.4.

For the reasons given in Section 6.2, the value of $n_c$ given by (6.2.2) has been used as well as the CPU time $t_c$ as a measure of computational labour. The values of $l_G$ in (6.2.2) for the different algorithms in this section are as follows.

For Algorithm 4.2.1 which is identical to Algorithm 3.1.2, $l_G = 3n$. For Algorithm 4.2.2, $l_G = 2n + 1$ because by assuming that one evaluation of $g$ is equivalent to about 3 units of computational labour, the evaluation of $k$ ($k \geqslant 1$) components of $g$ will be equivalent to about $1 + 2k/n$ units. Therefore, for Algorithm 4.2.2, we shall have

$$l_G = n + (2/n) \sum_{i=1}^{n} i = 2n + 1 .$$

For algorithms 4.3.1 (a) and 4.3.1 (b) $l_G = 3m$, where m is the number of non-zero diagonals of the Hessian. For algorithms 4.3.2 (a), 4.3.2 (b), 4.3.3 (a), 4.3.3 (b) $l_G = 3(m + 1)/2$.

Moreover, in the implementation of Algorithm 3.1.2, $\xi_1 = 2^{-28}$, $\eta = 0.9$, $\mu = 10^{-4}$, $\lambda = 1$ and the value of the finite-difference steplength h is given by $h = 2^{-28}$ as in Section 6.2.

Table 6.3.1 contains the results which are obtained when algorithms 4.2.1 and 4.2.2 are used to estimate G.

From Table 6.3.1, it would appear that Algorithm 4.2.2 is slightly superior to Algorithm 4.2.1 with regard to both function evaluations and computing time. Algorithms 4.2.1 and 4.2.2 were, in fact used to

solve all 28 test problems in Appendix 1 with similar results to those in Table 6.3.1.

From tables 6.3.2 and 6.3.3, it would appear that algorithms 4.3.2 (a) and 4.3.2 (b) are slightly superior to algorithms 4.3.1 (a) and 4.3.1 (b) respectively. Moreover, Algorithm 4.3.2 (b) requires more storage locations than algorithms 4.3.1, 4.3.2 (a) and 4.3.3.

From Table 6.3.4 it would appear that algorithms 4.3.3 (a) and 4.3.3 (b) are slightly superior to algorithms 4.3.1 (a) and 4.3.1 (b) respectively. Comparing the results for algorithms 4.3.2 (a) and 4.3.2 (b), with those for algorithms 4.3.3 (a) and 4.3.3 (b), it would appear that Algorithm 4.3.2 (a) is to be preferred to algorithms 4.3.2 (b), 4.3.3 (a) and 4.3.3 (b) on grounds of computing time. Furthermore, Algorithm 3.1.2 fails to solve Problem 1.27 when algorithms 4.3.1 (a) and 4.3.1 (b) are used, but is successful when algorithms 4.3.2 (a), 4.3.2 (b), 4.3.3 (a) and 4.3.3 (b) are used. Algorithm 3.1.2 fails to solve Problem 1.28 when any of the algorithms 4.3.1, 4.3.2 and 4.3.3 are used. Also, Algorithm 1.5.3 fails to solve this problem.

Clearly, the preceding conclusions rest upon results which have been obtained with a limited number, albeit quite large, of test problems. It may well be that there exist problems for which some or all of algorithms 4.2.2, 4.3.2 and 4.3.3 would fail to produce satisfactory estimates of G, while algorithms 4.2.1 and 4.3.1 are successful. The author has not yet found any such problems.

It should be noted that although in Table 6.3.1, Algorithm 4.2.2 requires three more evaluations of f and g and two more evaluations of G than does if Algorithm 4.2.1 to solve Problem 1.17, there is a very real saving in computing time if Algorithm 4.2.2 is used,

because in this case n = 9 and none of the components of g have any expressions in common. A similar situation exists in Table 6.3.2 for problems 1.3 and 1.13 which have related objective functions, save that in these cases, the components of g do have some common expressions, so that the saving in computing time is not so pronounced.

Algorithms 4.2.1 and 4.2.2 for the numerical estimation of the Hessian have been used with the minimization algorithms 3.1.2, 3.2.2 and 3.2.4, on test problems 1.1, 1.2, 1.4 - 1.9 and 1.16 - 1.22. It is found that Algorithm 4.2.2 is better than Algorithm 4.2.1 when used with Algorithm 3.1.2. Similar statements are valid when Algorithm 3.1.2 is replaced with Algorithm 3.2.2 and Algorithm 3.2.4 respectively. The relative performance of algorithms 3.1.2, 3.2.2, and 3.2.4 is unchanged by the substitution of Algorithm 4.2.2 in place of Algorithm 4.2.1.

Table 6.3.6 contains computational results corresponding to Algorithm 3.2.2 with p = 3, where Algorithm 4.2.2 is used to estimate the Hessian of the functions 1.1, 1.2, 1.4 - 1.9 and 1.16 - 1.22.

From tables 6.2.3 and 6.3.6 it may be concluded that, in terms of computing time, Algorithm 3.2.2 is superior to Algorithm 1.5.3 for all test problems save problems 1.16 and 1.18. Also, Algorithm 3.2.2 is significantly inferior to Algorithm 1.5.3 for problems 1.16 and 1.18. Also, from tables 6.2.4 and 6.3.6 it may be concluded that Algorithm 3.2.2 is more efficient than Algorithm 1.5.3 over fifteen test problems if we choose $t_c$ as an index of computational labour, but when $n_c$ is used as an index of computational labour, Algorithm 1.5.3 is still superior to Algorithm 3.2.2.

Algorithms 4.3.1, 4.3.2 and 4.3.3 for the numerical estimation of sparse Hessians have been used with the minimization algorithms

3.1.2, 3.2.2, and 3.2.4 on test problems 1.3, 1.10, 1.11 - 1.15 and 1.23 - 1.28. It is found that Algorithm 4.3.2 is better than algorithms 4.3.1 and 4.3.3 when used with Algorithm 3.1.2. Similar statements are valid when Algorithm 3.1.2 is replaced with Algorithm 3.2.2 and with Algorithm 3.2.4 respectively. The relative performance of algorithms 3.1.2, 3.2.2, and 3.2.4 is independent of the algorithm which is used to estimate the Hessian if this algorithm is chosen from algorithms 4.3.1 - 4.3.3.

Algorithm 1.5.3 has been used to solve the problems of Appendix 1 which have sparse Hessians and Table 6.3.5 contains the corresponding computational results.

From tables 6.3.2 - 6.3.5, it may be concluded that for all test problems Algorithm 1.5.3 is significantly inferior to Algorithm 3.1.2 when any of the algorithms 4.3.1 - 4.3.3 are used to estimate the Hessian in Algorithm 3.1.2.

| | Algorithm 4.2.1 | | | | | Algorithm 4.2.2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| PROBLEM | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
| 1.1 | 24 | 24 | 21 | 222 | 0.38 | 24 | 24 | 21 | 201 | 0.40 |
| 1.2 | 53 | 53 | 45 | 752 | 1.76 | 53 | 53 | 45 | 617 | 1.58 |
| 1.4 | 21 | 21 | 21 | 336 | 0.80 | 21 | 21 | 21 | 273 | 0.75 |
| 1.5 | 15 | 15 | 14 | 186 | 0.66 | 15 | 15 | 14 | 158 | 0.64 |
| 1.6 | 20 | 20 | 19 | 194 | 2.28 | 20 | 20 | 19 | 175 | 2.18 |
| 1.7 | 12 | 12 | 12 | 120 | 1.54 | 12 | 12 | 12 | 108 | 1.40 |
| 1.8 | 13 | 13 | 13 | 159 | 2.28 | 13 | 13 | 13 | 133 | 2.10 |
| 1.9 | 15 | 15 | 15 | 240 | 3.76 | 15 | 15 | 15 | 195 | 3.30 |
| 1.16 | 60 | 60 | 47 | 1086 | 28.94 | 60 | 60 | 47 | 851 | 26.84 |
| 1.17 | 28 | 28 | 28 | 868 | 564.22 | 31 | 31 | 30 | 694 | 373.58 |
| 1.18 | 13 | 13 | 13 | 276 | 75.82 | 13 | 13 | 13 | 211 | 50.22 |
| 1.19 | 55 | 55 | 48 | 1660 | 8.00 | 55 | 55 | 48 | 1228 | 7.42 |
| 1.20 | 30 | 30 | 30 | 1020 | 5.10 | 30 | 30 | 30 | 750 | 4.58 |
| 1.21 | 157 | 157 | 110 | 1948 | 17.08 | 157 | 157 | 110 | 1618 | 13.90 |
| 1.22 | 37 | 37 | 30 | 1048 | 4.44 | 37 | 37 | 30 | 778 | 3.60 |

Table 6.3.1

| | Algorithm 4.3.1 (a) | | | | | Algorithm 4.3.2 (a) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| PROBLEM | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
| 1.3 | 20 | 20 | 20 | 260 | 1.26 | 28 | 28 | 21 | 238 | 1.08 |
| 1.10 | 32 | 32 | 26 | 362 | 3.12 | 32 | 32 | 26 | 284 | 2.74 |
| 1.11 | 22 | 22 | 17 | 241 | 1.10 | 22 | 22 | 17 | 190 | 0.93 |
| 1.12 | 23 | 23 | 20 | 272 | 1.76 | 23 | 23 | 20 | 212 | 1.54 |
| 1.13 | 22 | 22 | 22 | 286 | 2.02 | 29 | 29 | 23 | 254 | 1.88 |
| 1.14 | 24 | 24 | 21 | 411 | 3.24 | 24 | 24 | 21 | 285 | 2.68 |
| 1.15 | 21 | 21 | 19 | 483 | 3.30 | 21 | 21 | 19 | 312 | 2.66 |
| 1.23 | 5 | 5 | 5 | 95 | 0.97 | 5 | 5 | 5 | 65 | 0.89 |
| 1.24 | 5 | 5 | 5 | 95 | 0.43 | 5 | 5 | 5 | 65 | 0.38 |
| 1.25 | 15 | 15 | 14 | 270 | 1.32 | 15 | 15 | 14 | 186 | 1.04 |
| 1.26 | 154 | 153 | 238 | 3155 | 10.18 | 55 | 54 | 43 | 475 | 2.00 |
| 1.27 | Failed | | | | | 108 | 107 | 82 | 921 | 5.96 |
| 1.28 | Failed | | | | | Failed | | | | |

Table 6.3.2

| PROBLEM | Algorithm 4.3.1 (b) | | | | | Algorithm 4.3.2 (b) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
| 1.3 | 20 | 20 | 20 | 260 | 1.26 | 28 | 28 | 21 | 238 | 1.08 |
| 1.10 | 32 | 32 | 26 | 362 | 3.16 | 32 | 32 | 26 | 184 | 2.82 |
| 1.11 | 22 | 22 | 17 | 241 | 1.08 | 22 | 22 | 17 | 190 | 0.96 |
| 1.12 | 23 | 23 | 20 | 171 | 1.78 | 23 | 23 | 20 | 212 | 1.60 |
| 1.13 | 22 | 22 | 22 | 286 | 1.98 | 28 | 28 | 22 | 244 | 1.82 |
| 1.14 | 24 | 24 | 21 | 411 | 3.22 | 24 | 24 | 21 | 285 | 2.80 |
| 1.15 | 21 | 21 | 19 | 483 | 3.28 | 21 | 21 | 19 | 312 | 2.82 |
| 1.23 | 5 | 5 | 5 | 95 | 1.02 | 5 | 5 | 5 | 65 | 0.86 |
| 1.24 | 5 | 5 | 5 | 95 | 0.44 | 5 | 5 | 5 | 65 | 0.38 |
| 1.25 | 15 | 15 | 14 | 270 | 1.38 | 15 | 15 | 14 | 186 | 1.06 |
| 1.26 | 254 | 253 | 238 | 3155 | 10.26 | 55 | 54 | 43 | 475 | 2.04 |
| 1.27 | Failed | | | | | 113 | 112 | 83 | 947 | 6.26 |
| 1.28 | Failed | | | | | Failed | | | | |

Table 6.3.3

| PROBLEM | Algorithm 4.3.3 (a) | | | | | Algorithm 4.3.3 (b) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
| 1.3 | 28 | 28 | 21 | 238 | 1.10 | 28 | 28 | 21 | 238 | 1.12 |
| 1.10 | 32 | 32 | 26 | 284 | 2.84 | 32 | 32 | 26 | 284 | 2.96 |
| 1.11 | 22 | 22 | 17 | 190 | 0.98 | 22 | 22 | 17 | 190 | 1.00 |
| 1.12 | 23 | 23 | 20 | 212 | 1.56 | 23 | 23 | 20 | 212 | 1.66 |
| 1.13 | 29 | 29 | 23 | 254 | 1.88 | 28 | 28 | 22 | 244 | 1.86 |
| 1.14 | 24 | 24 | 21 | 285 | 2.80 | 24 | 24 | 21 | 285 | 2.92 |
| 1.15 | 21 | 21 | 19 | 312 | 2.75 | 21 | 21 | 19 | 312 | 2.88 |
| 1.23 | 5 | 5 | 5 | 65 | 0.85 | 5 | 5 | 5 | 65 | 0.85 |
| 1.24 | 5 | 5 | 5 | 65 | 0.35 | 5 | 5 | 5 | 65 | 0.38 |
| 1.25 | 15 | 15 | 14 | 186 | 1.08 | 15 | 15 | 14 | 186 | 1.06 |
| 1.26 | 55 | 54 | 43 | 475 | 2.04 | 55 | 54 | 43 | 475 | 2.04 |
| 1.27 | 108 | 107 | 82 | 921 | 6.12 | 113 | 112 | 83 | 947 | 6.38 |
| 1.28 | Failed | | | | | Failed | | | | |

Table 6.3.4

| PROBLEM | $n_F$ | $n_g$ | $n_c$ | $t_c$ | Iteration |
|---|---|---|---|---|---|
| 1.3 | 47 | 47 | 188 | 2.36 | 46 |
| 1.10 | 95 | 95 | 380 | 9.98 | 83 |
| 1.11 | 38 | 38 | 152 | 2.28 | 34 |
| 1.12 | 57 | 57 | 228 | 4.96 | 51 |
| 1.13 | 88 | 88 | 352 | 6.32 | 88 |
| 1.14 | 74 | 74 | 296 | 8.84 | 68 |
| 1.15 | 72 | 72 | 188 | 8.14 | 66 |
| 1.23 | 26 | 26 | 104 | 2.54 | 18 |
| 1.24 | 15 | 15 | 60 | 0.76 | 11 |
| 1.25 | 35 | 35 | 140 | 11.52 | 31 |
| 1.26 | 89 | 88 | 353 | 3.54 | 71 |
| 1.27 | 229 | 229 | 916 | 12.92 | 183 |
| 1.28 | Failed | | | | |

Table 6.3.5

Algorithm 1.5.3
(Computational results due to Λ for the functions with sparse Hessian.)

| PROBLEM | $n_F$ | $n_g$ | $n_G$ | $n_c$ | $t_c$ |
|---|---|---|---|---|---|
| 1.1 | 44 | 42 | 16 | 248 | 0.42 |
| 1.2 | 85 | 73 | 30 | 574 | 1.52 |
| 1.4 | 37 | 37 | 12 | 256 | 0.62 |
| 1.5 | 23 | 21 | 11 | 163 | 0.64 |
| 1.6 | 23 | 20 | 9 | 128 | 1.50 |
| 1.7 | 14 | 13 | 5 | 78 | 1.00 |
| 1.8 | 14 | 12 | 7 | 99 | 1.46 |
| 1.9 | 25 | 19 | 10 | 172 | 2.90 |
| 1.16 | 79 | 71 | 34 | 734 | 23.80 |
| 1.17 | 23 | 23 | 8 | 244 | 136.86 |
| 1.18 | 23 | 22 | 8 | 193 | 44.30 |
| 1.19 | 69 | 68 | 25 | 798 | 4.90 |
| 1.20 | 35 | 34 | 12 | 389 | 2.46 |
| 1.21 | 275 | 241 | 94 | 1844 | 16.20 |
| 1.22 | 60 | 59 | 22 | 435 | 3.84 |
| Total | | | | 6355 | 242.42 |

Table 6.3.6

See pg 184.

(Computation results due to Algorithm 3.2.2 with p = 3 when the Hessian is approximated by Algorithm 4.2.2.)

## 6.4 Computational Results for the Least Squares Algorithms.

In this section, computational results corresponding to the algorithms contained in Chapter 5 are reported. Algorithm 1.2.2 has been used as the steplength algorithm. The bound $\lambda$ for the steplength required in Theorem 1.2.7 is taken to be $10^5$ for all test problems, save that $\lambda = 10$ for the problems 2.10, 2.16, 2.18 and 2.19 as recommended by Gill and Murray (1976). The values of $\epsilon_i (1 \leq i \leq 6)$ required in the algorithms of Chapter 5 are $\epsilon_1 = 2^{-28}$, $\epsilon_2 = 2^{-28}$, $\epsilon_3 = 10^{-6}$, $\epsilon_4 = 10^{-3}$, $\epsilon_5 = 10^{-6}$, $\epsilon_6 = \lambda$. Also in Algorithm 1.2.2, $\eta = 0.9$ and $\mu = 10^{-4}$.

From Section 6.2 and 6.3 we may conclude that the CPU time may be a more realistic measure of computational labour. Therefore, the comparison in this section is based upon $t_c$.

Table 6.4.1 contains the numerical results for algorithms 5.2.1 and 5.3.2 corresponding to those test problems of Appendix 2 for which the evaluation of B is required. Table 6.4.2 contains the numerical results for Algorithm 5.3.2 corresponding to the test problems for which results are not given in Table 6.4.1.

From Table 6.4.1 it may be concluded that for all of the test problems in Appendix 2, there is no significant difference in efficiency between algorithms 5.2.1 and 5.3.2 save for problems 2.2 7 and 2.2 8. This is probably because graded-Gauss-Newton steps have been taken seldom or not at all. The slight differences in efficiency are, however, noticeable in the solution of problems 2.4 and 2.12. When Algorithm 5.2.1 is used to solve Problem 2.4, for example, 37 iterations are required. The first 5 of these are Gauss-Newton iterations, iterations 6 and 7 are graded Gauss-Newton iterations, and iterations 8 - 16 are Newton iterations; the remainder

are Gauss-Newton iterations. When Algorithm 5.3.2 is used instead of Algorithm 5.2.1, 36 iterations are required. The first 5 of these are Gauss-Newton iterations, iterations 6 and 7 are graded Gauss-Newton iterations, and iterations 8 - 17 are Newton iterations; the remainder are Gauss-Newton iterations. Thus, for Problem 2.4 Algorithm 5.3.2 is more efficient than Algorithm 5.2.1 in terms of iteration number, time and evaluation of the function and the Jacobian, while for Problem 2.12, Algorithm 5.2.1 is more efficient with regard to computing time and number of evaluations of the function and the Jacobian. Moreover, Algorithm 5.2.1 fails to solve problems 2.27 and 2.28, but Algorithm 5.3.2 is successful on these problems, which suggests the superiority of Algorithm 5.3.2 over Algorithm 5.2.1.

For Algorithm 5.4.1, $p = 3$ gives almost optimal efficiency for all of the test problems of Appendix 2. The computational results corresponding to Algorithm 5.4.1 are contained in Table 6.4.3.

Algorithms 5.3.3, 5.5.1, and 5.6.1 give results which are distinct from those corresponding to Algorithm 5.3.2 only when the second derivative matrix B must be computed. For some problems of Appendix 2, B need never be computed.

Tables 6.4.4 and 6.4.5 contain the numerical results corresponding to algorithms 5.3.3, 5.5.1, and 5.6.1 respectively, for those problems, in which B is computed.

From tables 6.4.1, 6.4.2, and 6.4.3, it may be concluded that Algorithm 5.4.1 is superior to Algorithm 5.3.2 for 20 problems out of 32. For problems 2.27 and 2.28, Algorithm 5.4.1 failed, but Algorithm 5.3.2 was successful. Algorithms 5.4.1 and 5.3.2 are equally efficient on Problem 2.2.

From Table 6.4.4 the following conclusions may be drawn.

1. For 6 out of 32 test problems, Algorithm 5.3.3 is superior to Algorithm 5.6.1.
2. For problems 2.14 and 2.31, Algorithm 5.3.3 is significantly superior to Algorithm 5.6.1.
3. For problems 2.27, and 2.28, Algorithm 5.3.3 failed to find any solution, while Algorithm 5.6.1 found a local minimizer.

From tables 6.4.3 and 6.4.5 the following conclusions may be drawn.

1. Algorithm 5.4.1 is superior to Algorithm 5.5.1 for 21 problems out of 32.
2. For problems 2.27 and 2.28 Algorithm 5.4.1 could decrease the function substantially, but Algorithm 5.5.1 failed at an early stage.

From tables 6.4.1 and 6.4.5 the following conclusions may be drawn.

1. For 6 out of 32 problems, Algorithm 5.5.1 is superior to Algorithm 5.3.2.
2. For problems 2.27 and 2.28 Algorithm 5.5.1 failed, while Algorithm 5.3.2 did not.

As has been mentioned previously, algorithms 5.3.2, 5.5.1, and 5.6.1 are equally efficient for those problems for which B is not computed.

The total computing time obtained by using algorithms 5.3.2, 5.5.1 and 5.6.1 to solve the problems indicated in Table 6.4.4 with the exception of problems 2.14, 2.27, 2.28 and 2.31, is contained in Table 6.4.6.

From Table 6.4.6 it may be concluded that Algorithm 5.4.1 is the best over 13 test problems and Algorithm 5.5.1 is the second best.

Since Algorithm 5.5.1 requires m arrays each of size n (n + 1)/2 to evaluate the Hessians of the components of f in addition to the storage space required, then Algorithm 5.3.2 may be preferred for large n. Algorithm 5.6.1 is superior to Algorithm 5.3.3 and the saving in computing time is about 8%.

| | Algorithm 5.2.1 | | | | | Algorithm 5.3.2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| PROBLEM | $n_f$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ | $n_f$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ |
| 2.4 | 79 | 79 | 11 | 9.80 | 37 | 73 | 73 | 12 | 9.44 | 36 |
| 2.6 | 24 | 24 | 1 | 1.52 | 12 | 24 | 24 | 1 | 1.34 | 12 |
| 2.12 | 18 | 18 | 4 | 0.58 | 8 | 24 | 24 | 4 | 0.64 | 4 |
| 2.13 | 10 | 10 | 3 | 12.94 | 7 | 10 | 10 | 3 | 12.74 | 7 |
| 2.14 | 46 | 46 | 7 | 16.70 | 22 | 46 | 46 | 7 | 16.56 | 22 |
| 2.15 | 6 | 6 | 1 | 1.40 | 5 | 6 | 6 | 1 | 1.32 | 5 |
| 2.16 | 35 | 35 | 6 | 1.88 | 10 | 35 | 35 | 6 | 1.96 | 10 |
| 2.17 | 11 | 11 | 3 | 2.94 | 8 | 11 | 11 | 3 | 2.90 | 8 |
| 2.18 | 12 | 12 | 1 | 10.24 | 8 | 12 | 12 | 1 | 10.32 | 8 |
| 2.19 | 19 | 19 | 3 | 121.04 | 11 | 19 | 19 | 3 | 121.66 | 11 |
| 2.20 | 12 | 12 | 4 | 0.60 | 10 | 12 | 12 | 4 | 0.62 | 10 |
| 2.22 | 11 | 11 | 4 | 1.42 | 8 | 11 | 11 | 4 | 1.38 | 8 |
| 2.27 | | | Failed | | | 229 | 229 | 101 | 32.58 | 139 |
| 2.28 | | | Failed | | | 232 | 232 | 104 | 32.82 | 138 |
| 2.29 | 20 | 20 | 6 | 4.36 | 12 | 20 | 20 | 6 | 4.18 | 12 |
| 2.30 | 10 | 10 | 2 | 2.70 | 7 | 10 | 10 | 2 | 2.68 | 7 |
| 2.31 | 26 | 26 | 3 | 4.52 | 7 | 26 | 26 | 3 | 4.56 | 7 |

Table 6.4.1

Algorithm 5.3.2

| PROBLEM | $n_f$ | $n_J$ | $n_E$ | $t_c$ | $n_I$ |
|---|---|---|---|---|---|
| 2.1 | 32 | 32 | 0 | 0.68 | 13 |
| 2.2 | 15 | 15 | 0 | 1.10 | 10 |
| 2.3 | 18 | 18 | 0 | 3.06 | 17 |
| 2.5 | 6 | 6 | 0 | 0.26 | 3 |
| 2.7 | 6 | 6 | 0 | 0.18 | 3 |
| 2.8 | 14 | 14 | 0 | 0.42 | 7 |
| 2.9 | 18 | 18 | 0 | 3.16 | 16 |
| 2.10 | 8 | 8 | 0 | 1.56 | 6 |
| 2.11 | 5 | 5 | 0 | 1.20 | 2 |
| 2.21 | 6 | 6 | 0 | 17.02 | 5 |
| 2.23 | 34 | 34 | 0 | 0.68 | 13 |
| 2.24 | 8 | 8 | 0 | 0.82 | 5 |
| 2.25 | 5 | 5 | 0 | 1.34 | 4 |
| 2.26 | 6 | 6 | 0 | 1.28 | 5 |
| 2.32 | 6 | 6 | 0 | 24.30 | 5 |

Table 6.4.2

Algorithm 5.4.1 p = 3

| PROBLEM | $n_f$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ | PROBLEM | $n_f$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2.1 | 34 | 31 | 0 | 0.62 | 12 | 2.21 | 11 | 5 | 1 | 16.48 | 4 |
| 2.2 | 17 | 13 | 0 | 1.10 | 10 | 2.22 | 11 | 8 | 4 | 1.28 | 7 |
| 2.3 | 31 | 11 | 0 | 1.98 | 10 | 2.23 | 33 | 31 | 0 | 0.70 | 12 |
| 2.4 | 90 | 70 | 15 | 10.20 | 36 | 2.24 | 10 | 8 | 1 | 0.86 | 5 |
| 2.5 | 4 | 2 | 0 | 0.12 | 1 | 2.25 | 7 | 5 | 0 | 1.36 | 4 |
| 2.6 | 23 | 13 | 0 | 1.10 | 7 | 2.26 | 8 | 5 | 0 | 0.98 | 4 |
| 2.7 | 7 | 4 | 0 | 0.16 | 3 | 2.27 | 14 | 12 | 4 | 1.40 | 5* |
| 2.8 | 15 | 11 | 0 | 0.40 | 6 | 2.28 | 13 | 12 | 5 | 1.58 | 6* |
| 2.9 | 30 | 12 | 0 | 2.20 | 10 | 2.29 | 32 | 26 | 12 | 6.52 | 18 |
| 2.10 | 11 | 7 | 0 | 1.36 | 5 | 2.30 | 11 | 5 | 0 | 1.64 | 4 |
| 2.11 | 7 | 5 | 0 | 1.26 | 2 | 2.31 | 18 | 9 | 3 | 2.44 | 6 |
| 2.12 | 41 | 34 | 5 | 0.74 | 10 | 2.32 | 10 | 5 | 0 | 21.78 | 4 |
| 2.13 | 14 | 7 | 3 | 12.60 | 6 | | | | | | |
| 2.14 | 53 | 46 | 7 | 17.16 | 22 | | | | | | |
| 2.15 | 10 | 5 | 1 | 1.22 | 4 | | | | | | |
| 2.16 | 31 | 25 | 5 | 1.48 | 8 | | | | | | |
| 2.17 | 15 | 12 | 3 | 3.02 | 8 | | | | | | |
| 2.18 | 16 | 10 | 2 | 9.52 | 7 | | | | | | |
| 2.19 | 24 | 19 | 3 | 114.22 | 10 | | | | | | |
| 2.20 | 23 | 13 | 3 | 0.60 | 9 | | | | | | |

Table 6.4.3

(*) convergence criteria not satisfied.

| | Algorithm 5.3.3 | | | | | Algorithm 5.6.1 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| PROBLEM | $n_f$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ | $n_F$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ |
| 2.4 | 73 | 116 | | 10.30 | 36 | 77 | 77 | | 10.00 | 37 |
| 2.6 | 24 | 29 | | 1.42 | 12 | 24 | 24 | | 1.44 | 12 |
| 2.12 | 24 | 30 | | 0.62 | 9 | 31 | 31 | | 0.84 | 14 |
| 2.13 | 10 | 23 | | 16.74 | 7 | 9 | 9 | | 13.78 | 8 |
| 2.14 | 46 | 67 | | 19.54 | 22 | 253 | 253 | | 89.30 | 116(1) |
| 2.15 | 6 | 7 | | 1.42 | 5 | 6 | 6 | | 1.38 | 5 |
| 2.16 | 35 | 43 | | 1.94 | 10 | 43 | 43 | | 2.26 | 13 |
| 2.17 | 11 | 21 | | 3.42 | 8 | 12 | 12 | | 3.16 | 9 |
| 2.18 | 12 | 13 | | 10.32 | 8 | 12 | 12 | | 10.12 | 8 |
| 2.19 | 18 | 32 | | 135.50 | 10 | 20 | 20 | | 128.54 | 12 |
| 2.20 | 12 | 19 | | 0.60 | 10 | 15 | 15 | | 0.72 | 13 |
| 2.22 | 9 | 20 | | 1.68 | 8 | 10 | 10 | | 1.62 | 9 |
| 2.27 | Failed | | | | | 9 | 9 | | 0.52 | 4** |
| 2.28 | Failed | | | | | 9 | 9 | | 0.54 | 4** |
| 2.29 | 20 | 38 | | 5.34 | 12 | 21 | 21 | | 4.08 | 13 |
| 2.30 | 10 | 15 | | 3.18 | 7 | 14 | 14 | | 3.12 | 7 |
| 2.31 | 26 | 34 | | 5.20 | 7 | 58 | 58 | | 11.06 | 18 |

Table 6.4.4

(1). The sum of squares obtained at iteration 15, agrees with the sum at the minimum with 5 decimal places, but, the convergence criteria were not satisfied.

(**) local minimum has been found.

Algorithm 5.5.1

| PROBLEM | $n_F$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ |
|---|---|---|---|---|---|
| 2.4 | 84 | 84 | 5 | 8.64 | 31 |
| 2.6 | 24 | 24 | 1 | 1.46 | 12 |
| 2.12 | 24 | 24 | 4 | 0.64 | 9 |
| 2.13 | 7 | 7 | 2 | 12.28 | 6 |
| 2.14 | 46 | 46 | 7 | 18.02 | 22 |
| 2.15 | 6 | 6 | 1 | 1.46 | 5 |
| 2.16 | 35 | 35 | 5 | 2.02 | 9 |
| 2.17 | 13 | 13 | 5 | 4.24 | 10 |
| 2.18 | 12 | 12 | 1 | 10.66 | 8 |
| 2.19 | 18 | 18 | 2 | 117.96 | 10 |
| 2.20 | 12 | 12 | 4 | 0.64 | 10 |
| 2.22 | 7 | 7 | 2 | 1.16 | 6 |
| 2.27 | Failed | | | | |
| 2.28 | Failed | | | | |
| 2.29 | 16 | 16 | 2 | 3.06 | 8 |
| 2.30 | 10 | 10 | 2 | 3.00 | 7 |
| 2.31 | 9 | 9 | 2 | 2.64 | 6 |

Table 6.4.5

| | Algorithm | | | | |
|---|---|---|---|---|---|
| Algorithm | 5.3.2 | 5.3.3 | 5.4.1 | 5.5.1 | 5.6.1 |
| $T_c$ | 171.18 | 197.68 | 164.14 | 167.22 | 181.06 |

Table 6.4.6

(Total computing time over the problems 2.4, 2.6, 2.12, 2.13, 2.15 - 2.20, 2.22, 2.29 and 2.30)

## 6.5 Computational Results Corresponding to the Use of Powell's Steplength Algorithm.

In Section 1.6, the general minimization algorithm due to Powell has been given. This contains a simple procedure for finding the steplength $\alpha^{(K)}$ in such a way that the sequence generated from the algorithm converges to a critical point of the objective function. The efficiency of Powell's steplength algorithm may be compared with that of Algorithm 1.2.2 by using Powell's steplength algorithm in algorithms 3.1.2 and 5.3.2 in place of Algorithm 1.2.2 to solve the test problems listed in appendices 1 and 2 respectively. The computational results contained in tables 6.5.1 and 6.5.2 correspond to $\bar{\Delta} = \lambda$ (where $\lambda$ is as in Algorithm 1.2.2), $C_1 = 10^{-4}$, $C_2 = 2$ and $C_3 = 1/2$.

From tables 6.2.1 and 6.5.1 it may be concluded that Powell's steplength algorithm increases the efficiency of Algorithm 3.1.2 for 6 out of 15 test problems. Also, from Table 6.5.3 it may be concluded that Powell's steplength algorithm requires more computing time than Algorithm 1.2.2 when used with Algorithm 3.1.2.

From tables 6.5.2, 6.4.1 and 6.4.2, the saving in computing time obtained by using Powell's steplength algorithm is not significant compared with that obtained by using Algorithm 1.2.2 as steplength algorithm. Moreover, Algorithm 5.3.2 fails for problems 2.27 and 2.28 and a large number of iterations are required for problems 2.16, 2.29 and 2.31 when Powell's steplength algorithm is used. This suggests that Powell's steplength algorithm may be no more effective than Algorithm 1.2.2, in general.

| PROBLEM | $n_F$ | $n_g$ | $n_G$ | $t_c$ |
|---|---|---|---|---|
| 1.1 | 21 | 21 | 21 | 0.36 |
| 1.2 | 45 | 45 | 45 | 1.82 |
| 1.4 | 21 | 21 | 21 | 0.88 |
| 1.5 | 14 | 14 | 14 | 0.70 |
| 1.6 | 19 | 19 | 19 | 2.22 |
| 1.7 | 12 | 12 | 12 | 1.48 |
| 1.8 | 13 | 13 | 13 | 2.34 |
| 1.9 | 15 | 15 | 15 | 3.80 |
| 1.16 | 48 | 45 | 45 | 19.22 |
| 1.17 | 29 | 29 | 29 | 587.16 |
| 1.18 | 13 | 13 | 13 | 76.14 |
| 1.19 | 48 | 48 | 48 | 9.94 |
| 1.20 | 31 | 31 | 31 | 6.46 |
| 1.21 | 108 | 108 | 108 | 15.12 |
| 1.22 | 31 | 31 | 31 | 4.38 |

Table 6.5.1

(Computational Results corresponding to Algorithm 3.1.2 with Powell's Steplength Algorithm)

| PROBLEM | $n_f$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ | PROBLEM | $n_f$ | $n_J$ | $n_B$ | $t_c$ | $n_I$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2.1 | 22 | 22 | 0 | 0.56 | 11 | 2.21 | 8 | 8 | 0 | 23.90 | 7 |
| 2.2 | 12 | 12 | 0 | 1.10 | 10 | 2.22 | 11 | 11 | 4 | 1.56 | 8 |
| 2.3 | 18 | 18 | 0 | 3.40 | 17 | 2.23 | 24 | 24 | 0 | 0.60 | 12 |
| 2.4 | 55 | 55 | 16 | 11.02 | 40 | 2.24 | 7 | 7 | 1 | 0.82 | 5 |
| 2.5 | 4 | 4 | 0 | 0.32 | 3 | 2.25 | 5 | 5 | 0 | 1.32 | 4 |
| 2.6 | 14 | 14 | 0 | 1.40 | 10 | 2.26 | 6 | 6 | 0 | 1.26 | 5 |
| 2.7 | 4 | 4 | 0 | 0.20 | 3 | 2.27 | Failed | | | | |
| 2.8 | 14 | 14 | 0 | 0.46 | 8 | 2.28 | Failed | | | | |
| 2.9 | 18 | 18 | 0 | 3.38 | 16 | 2.29 | 1015 | 1015 | 99 | 102.60 | 103 |
| 2.10 | 7 | 7 | 0 | 1.54 | 6 | 2.30 | 8 | 8 | 1 | 2.42 | 6 |
| 2.11 | 24 | 24 | 0 | 2.52 | 2 | 2.31 | 1015 | 1015 | 50 | 118.60 | 54 |
| 2.12 | 16 | 16 | 4 | 0.56 | 9 | 2.32 | 9 | 9 | 0 | 38.92 | 8 |
| 2.13 | 9 | 9 | 3 | 13.88 | 7 | | | | | | |
| 2.14 | 50 | 50 | 9 | 19.32 | 25 | | | | | | |
| 2.15 | 6 | 6 | 1 | 1.40 | 5 | | | | | | |
| 2.16 | 232 | 232 | 103 | 17.22 | 114 | | | | | | |
| 2.17 | 13 | 13 | 2 | 3.36 | 9 | | | | | | |
| 2.18 | 12 | 12 | 2 | 10.62 | 8 | | | | | | |
| 2.19 | 16 | 16 | 3 | 111.64 | 10 | | | | | | |
| 2.20 | 12 | 12 | 4 | 0.60 | 10 | | | | | | |

Table 6.5.2

(Computational Results corresponding to Algorithm 5.3.2 with Powell's Steplength Algorithm)

| | Algorithm 1.2.2 | Powell's steplength Algorithm |
|---|---|---|
| $T_c$ Sec. | 717.06 | 732.02 |

Table 6.5.3

## 6.6 Comparison of Algorithms 3.1.1 and 3.1.2 with the Algorithm of Fletcher and Freeman.

Fletcher and Freeman (1977) have described a Newton-type method for unconstrained optimization which they claim may be more efficient than Algorithm 3.1.1 of Gill and Murray. Therefore it would seem to be desirable to compare the algorithms of Fletcher and Freeman and Gill and Murray. Table 6.6.1 contains the results which were obtained by using an implementation of the algorithm of Fletcher and Freeman which was provided by Fletcher. The Hessian has been estimated numerically as in Algorithm 3.1.2. Also, Table 6.6.2 contains the computing time corresponding to the algorithm of Fletcher and Freeman when the Hessian is computed analytically.

From tables 6.2.1, 6.2.7, 6.6.1 and 6.6.2, it may be concluded that the Newton-type algorithm of Gill and Murray is more reliable and efficient than the Newton-type algorithm of Fletcher and Freeman (1977), at least for the problems indicated.

| PROBLEM | $n_F$ | $n_g$ | $n_H$ | $t_c$ | |
|---|---|---|---|---|---|
| 1.1 | 35 | 35 | 14 | 0.30 | |
| 1.2 | 66 | 66 | 24 | 1.16 | |
| 1.4 | 41 | 41 | 20 | 0.86 | |
| 1.5 | 33 | 33 | 11 | 0.78 | |
| 1.6 | 34 | 34 | 10 | 2.16 | |
| 1.7 | 22 | 22 | 8 | 1.62 | |
| 1.8 | 22 | 22 | 9 | 2.32 | |
| 1.9 | 36 | 36 | 13 | 4.56 | |
| 1.16 | Failed (overflow) | | | | |
| 1.17 | After 15 minutes the solution had not been found. | | | | |
| 1.18 | 10 | 10 | 1 | 16.20 | Failed |
| 1.19 | 1000 | 1000 | 173 | 39.48 | |
| 1.20 | 1000 | 1000 | 166 | 37.76 | |
| 1.21 | 224 | 244 | 21 | 9.58 | Convergence criteria not satisfied. |
| 1.22 | 141 | 141 | 14 | 6.12 | |

Table 6.6.1

(Computational Results corresponding to the modified Newton method of Fletcher and Freeman)

| PROBLEM | $t_c$ |
|---|---|
| 1.1 | 0.30 |
| 1.2 | 0.82 |
| 1.3 | 1.28 |
| 1.4 | 0.66 |
| 1.5 | 0.64 |
| 1.6 | 2.10 |
| 1.7 | 1.54 |
| 1.8 | 1.94 |
| 1.9 | 3.98 |
| 1.10 | 3.28 |

Table 6.6.2

(Computing time corresponding to the algorithm of Fletcher and Freeman (1977), when analytic Hessian is available.)

## 6.7 Comparison of Algorithms 5.2.1 and 5.3.2 with Fletcher's Least Squares Algorithm.

Fletcher (1971) has described an efficient implementation of Marquardt's method for the least squares problem. Therefore, it would seem to be desirable to compare algorithms 5.2.1 and 5.3.2 with Fletcher's least squares algorithm. Table 6.7.1 contains the results which were obtained by using an implementation of the algorithm of Fletcher which is available in the NAG library.

From tables 6.4.1, 6.4.2 and 6.7.1, it may be concluded that Fletcher's algorithm is more efficient than algorithms 5.2.1 and 5.3.2 for small residual problems, while Algorithm 5.3.2 is more reliable than Fletcher's algorithm for large residual problems. This can be seen by comparing these algorithms for problems 2.14, 2.18, 2.20 and 2.29 - 2.31. (See tables 6.4.1 and 6.7.1)

From tables 6.7.1 and 6.4.4, it may be concluded that Algorithm 5.6.1 is more reliable but less efficient, than Fletcher's implementation of the Levenberg-Marquardt algorithm.

| PROBLEM | $n_f$ | $n_J$ | $t_c$ | $n_I$ | PROBLEM | $n_f$ | $n_J$ | $t_c$ | $n_I$ |
|---|---|---|---|---|---|---|---|---|---|
| 2.1 | 18 | 14 | 0.32 | 14 | 2.21 | 7 | 6 | 5.58 | 6 |
| 2.2 | 12 | 9 | 0.46 | 9 | 2.22 | 28 | 23 | 1.42 | 23 |
| 2.3 | 29 | 28 | 1.46 | 28 | 2.23 | 20 | 15 | 0.36 | 15 |
| 2.4 | 71 | 64 | 3.92 | 64 | 2.24 | 18 | 13 | 0.78 | 13 |
| 2.5 | 3 | 2 | 0.08 | 2 | 2.25 | 6 | 5 | 0.54 | 5 |
| 2.6 | 19 | 15 | 0.66 | 15 | 2.26 | 7 | 6 | 0.50 | 6 |
| 2.7 | 3 | 2 | 0.04 | 2 | 2.27 | 88 | 58 | 5.26 | 58 |
| 2.8 | 15 | 12 | 0.26 | 12 | 2.28 | 87 | 57 | 5.26 | 57 |
| 2.9 | 44 | 43 | 2.64 | 43 | 2.29 | 174 | 133 | 17.68 | 133 |
| 2.10 | 7 | 6 | 0.64 | 6 | 2.30 | 109 | 108 | 15.02 | 108 |
| 2.11 | 3 | 2 | 0.26 | 2 | 2.31 | 120 | 119 | 16.52 | 119* |
| 2.12 | 21 | 18 | 0.40 | 18 | 2.32 | 7 | 6 | 7.64 | 6 |
| 2.13 | 12 | 10 | 5.32 | 10 | | | | | |
| 2.14 | 478 | 411 | 99.14 | 411 | | | | | |
| 2.15 | 8 | 7 | 0.50 | 7 | | | | | |
| 2.16 | 20 | 12 | 0.80 | 12 | | | | | |
| 2.17 | 23 | 20 | 1.86 | 20 | | | | | |
| 2.18 | 168 | 118 | 47.30 | 118 | | | | | |
| 2.19 | 17 | 15 | 29.08 | 15 | | | | | |
| 2.20 | 43 | 41 | 0.96 | 41 | | | | | |

Table 6.7.1

(Computational Results corresponding to Fletcher's Least Squares algorithm.)

(*) convergence criteria not satisfied.

## 6.8 Conclusions

If analytical formulae for the components of the gradient of an objective function are available, then from sections 6.2, 6.3 and 6.6 the following conclusions may be drawn.

1. If analytic formulae for the Hessian of the objective function are not known then
   (a) Algorithm 3.2.2 with $p = 3$ would appear to be the best of algorithms 3.1.2, 3.2.2, 3.2.4 - 3.2.7 and 1.5.3 when the Hessian matrix is full and Algorithm 4.2.2 is used to estimate its value.
   (b) When the Hessian matrix is full and the gradient of the objective function is computationally expensive then Algorithm 3.2.5 with $p = 3$, $\varepsilon_4 = 10^{-1}$ would appear to be the best of algorithms 3.1.2, 3.2.2, 3.2.4 - 3.2.7, and 1.5.3.
   (c) If the Hessian is an m diagonal matrix, then Algorithm 3.2.2 with $p = 3$ is the most efficient of algorithms 3.1.2, 3.2.2, 3.2.4 - 3.2.7, and 1.5.3 if Algorithm 4.3.2 is used to estimate the Hessian matrix.
2. When the Hessian of the objective function is computed analytically, Algorithm 3.3.3 with $p = 2$ would appear to be the most efficient and reliable of algorithms 3.1.1, and 3.3.1 - 3.3.6.

For both the general minimization and least squares algorithms, Algorithm 1.2.2 appears to be preferable to Powell's steplength algorithm.

For the least squares algorithms discussed in sections 6.4, and 6.7, the following conclusions may be drawn.

1. When B is computed analytically, Algorithm 5.4.1 is most efficient, while Algorithm 5.3.2 is more reliable algorithm than algorithms 5.2.1, and 5.4.1.
2. If B is not available analytically then Algorithm 5.6.1 is more reliable than the least squares algorithms 5.3.3, and Fletcher's algorithm. Algorithm 5.6.1 often requires more computing time than does Fletcher's algorithm; however, this is because singular value decomposition is a time-consuming procedure. Therefore, in this case, the most reliable of the three least squares algorithms would appear to be Algorithm 5.3.2, while the most efficient would appear to be Fletcher's algorithm.

## References


1. Altman, M., 1966, Generalized Gradient Methods of Minimizing a Functional, Bul. Acad. Polon. Sci., Ser., Sci. Math. Astronom. Phys., 14, 313-318.
2. Armijo, L., 1966, Minimization of Functions having Lipschitz Continuous First Partial Derivatives, Pacific J. Math. 16, 1-3.
3. Avriel, M., 1976, Nonlinear Programming Analysis and Methods, Prentice-Hall Series in Automatic Computation.
4. Bard, Y., 1968, On a Numerical Instability of Davidon-like Methods, Math. Comp. 22, 665-666.
5. Bard, Y., 1970, Comparison of Gradient Methods for the Solution of Non-linear Parameter Estimation Problems. SIAM. Numer. Anal. 7, 157-186.
6. Beale, E.M.L., 1958, On an Iterative Method for Finding a Local Minimum of a Function of more than One Variable, Princeton University, Princeton, New Jersey, Statistical Techniques Research Group, Technical Report No. 25.
7. Betts, J.T., 1976, Solving the Nonlinear Least Square Problem, Application of a General Method. J.O.T.A. 18, 469-484.
8. Biggs, M.C., 1977, Minimization Algorithms Making Use of Non-quadratic Properties of the Objective Function, J. Inst. Math. Appl. 8, 315-327.
9. Box, M., 1966, A Comparison of Several Current Optimization Methods and the Use of Transformations in Constrained Problems, Comp. J., 9, 67-77.
10. Bosarge, W. E., and Falb, P.L., 1969, A Multipoint Method of Third Order, J.O.T.A, 4, 156-166.

11. Bosarge, W.E., and Falb, P.L., 1970, Infinite Dimensional Multi-point Methods and the Solution of Two Point Boundary Value Problems, Numer. Math. 14, 264-286.

12. Branin, F.H. Jr., 1971, Widely Convergent Method for Finding Multiple Solutions of Simultaneous Nonlinear Equations. IBM. Journal, Research and Development 16, 504-522.

13. Brent, R.P., 1973, Algorithms for Minimization Without Derivatives. Prentice-Hall, Englewood Cliffs, New Jersey.

14. Brodlie, K.W., 1977, Unconstrained Minimization in "The State of the Art in Numerical Analysis" (Ed.) Jacobs, D.A.H.

15. Broyden, C.G., 1965, A Class of Methods for Solving Nonlinear Simultaneous Equations, Math. Comp. 19, 557-593.

16. Broyden, C.G., 1967, Quasi-Newton Methods and their Application to Function Minimization, Math. Comp. 21, 368-381.

17. Broyden, C.G., 1970, The Convergence of a Class of Double-Rank Minimization Algorithm : 1. General Considerations; J. Inst. Math. Appl. 6, 76-90.

18. Broyden, C.G., Dennis, J.E., and More, J.J., 1973, On the Local and Superlinear Convergence of Quasi-Newton Methods, J. Inst. Math. Appl. 12, 223-245.

19. Cauchy, A., 1829, Sur la Determination Approximative des Raines d'une equation Algebraic on transcendante, CEURVRES, Complete (II), 4, 573-609, Gaunthier Villars, Paris.

20. Cauchy, A., 1847, Method General Pour la resolution des Systems d'equations Simultanes, C.R. Acad. Sc., Paris, 25, 536-538.

21. Coggin, G.F., 1964, Univariate Search Methods, Imperial Industries Ltd. Central Instr. Lab. Research.

22. Courant, R., 1943, Variational Methods for the Solution of

Problems of Equilibrium and Variations, Bull. Amer. Math. SOC. 49.

23. Cragg, E.E. and Levy, A.V., 1969, Study on a Supermemory Gradient Method for the Minimization of Functions. J.O.T.A. 4, 191-205.

24. Crockett, J.B., and Chernoff, H., 1955, Gradient Methods of Maximization, Pacific, J. Math. 5, 33-50.

25. Curry, H.B., 1944, The Method of Steepest Descent of Nonlinear Minimization Problems, Quart. J. Appl. Math. 2, 258-261.

26. Davidon, W.C., 1959, Variable Metric Method for Minimization, Research and Development, Report ANL - 5990, US Atomic Commission, Argonne National Laboratories.

27. Davidon, W.C., 1975, Optimally Conditioned Optimization Algorithms Without Line Searches, Math. Prog. 9, 1-30.

28. Davidon, W.C., 1976, New Least Square Algorithms. J.O.T.A. 18, 187-198.

29. Dennis, J.E., and Moré, J.J., 1974, A Characterization of Super-linearly Convergence and its Application to Quasi-Newton Methods, Math. Comp., 28, 549-560.

30. Dennis, J.E. Jr., and Moré, J.J., 1977, Quasi-Newton Methods, Motivation and Theory, SIAM, Review, 9, 46-89.

31. Dixon, L.C.W., 1972, Quasi-Newton Algorithms Generate Identical Points. Math. Prog. 2 (3), 393-388.

32. Dixon, L.C.W., 1973, Nonlinear Optimization: A Survey of the Art, The Hatfield Polytechnic, Numerical Optimization Centre, T.R. 42.

33. Dixon, L.C.W., 1973a, Conjugate Directions Without Linear Searches, J. Inst. Math. Appl. 11, 317-328.

34. Dixon, L.C.W., and Biggs, M.C., 1970, MEANDER -, A Newton Based

Procedure for n-dimensional Minimization, the Hatfield Polytechnic, Numerical Optimization Centre, T.R.9.

35. Engwall, J.L., 1966, Numerical Algorithms for Solving Overdetermined Systems of Non-linear Equations. NASA document N70-35600.

36. Fiacco, A.V., and McCormick, G.P., 1968, Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Willey and Sons, Inc.

37. Fletcher, R., 1970, A New Approach to Variable Metric Algorithms, Comp. J. 13, 317-322.

38. Fletcher, R., and Freeman, T.L., 1977, A Modified Newton Method Minimization, J.O.T.A., 23, 357-372.

39. Fletcher, R., 1972, A Survey of Algorithms for Unconstrained Optimization, in "Numerical Methods for Unconstrained Optimization" Murray (Ed.).

40. Fletcher, R., and Powell, M.J.D., 1963, A Rapidly Convergent Descent Method for Minimization, Comp. J., 7, 163-188.

41. Freudenstein, F., and Roth, B., 1963, Numerical Solutions of Nonlinear Equations, J. ACM, 10, 550-556.

42. Gill, P.E., and Murray, W., 1972, Quasi-Newton Methods for Unconstrained Optimization, J. Inst. Math. Appl., 9, 91-108.

43. Gill, P.E., and Murray, W., 1974, Numerical Methods for Constrained Optimization (Ed.) Academic Press.

44. Gill, P.E., and Murray, W., 1974a, Newton-type Methods for Unconstrained and Linearly Constrained Optimization, Math. Prog. 7, 311-350.

45. Gill, P.E., and Murray, W., 1974b, Safeguarded Steplength Algorithms for Optimization Using Descent Methods, National Physical

Laboratory, Report NAC 37.

46. Gill, P.E., and Murray, W., 1974c, Methods for Large-scale Linearly Constrained Problems in "Numerical Methods for Constrained Optimization" (Academic Press).

47. Gill, P.E., and Murray, W., 1975, "Nonlinear Least Squares and Nonlinearly Constrained Optimization" in Numerical Analysis, Dundee, 1975, Sprin-Verlag Lecture Notes in Maths 506.

48. Gill, P.E., and Murray, W., 1976, Algorithms for the Solution of the Nonlinear Least-Square Problem, National Physical Laboratory, Report, NAC 71.

49. Gill, P.E., Murray, W., and Picken, S.M., 1974, The Implementation of Two Modified Newton Algorithms for Unconstrained Optimization, National Physical Laboratory Report NAC 24.

50. Gill, P.E., Murray, W., and Pitfield, R.A. 1972, The Implementation of Two Revised Quasi-Newton Algorithms for Unconstrained Optimization, National Physical Laboratory Report NAC 11.

51. Glayzal, A., 1959, Solution of Nonlinear Equations, Quart. J. Appl. Math. 17, 95-96.

52. Goldfarb, D., 1970, A Family of Variable Metric Methods Derived by Variational Means, Math Comp. 24, 23-26.

53. Goldstein, A.A., 1962, Cauchy's Method of Minimization, Numer. Math. 4, 146-150.

54. Goldstein, A.A., 1967, Constructive Real Analysis, Harper and Row, New York.

55. Goldstein, A.A., and Price, J., 1967, An Effective Algorithm for Minimization, Numer. Math. 10, 184-189.

56. Greenstadt, J., 1967, On the Relative Efficiencies of Gradient Methods, Math. Comp. 21, 360-367.

57. Hartley, H.O., 1961, The Modified Gauss-Newton Method for the Fitting of Nonlinear Regression Functions by Least Squares, Technometrics 3, 269-280.

58. Himmelblau, D.M., 1972, Applied Nonlinear Programming, McGrow Hill Book Company.

59. Huang, H.Y., 1970, Unified Approach to Quadratically Convergent Algorithms for Function Minimization, J.O.T.A., 5, 405-423.

60. Jennrich, R.I., and Sampson, P.F., 1968, Application of Stepwise Regression to Nonlinear Estimation. Technometrics, 10, No. 1.

61. Johnson, S.M., 1955, Best Extrapolation for Maximum in Fibonaccian, The RAND Corporation, RM - 1590.

62. Kantorovich, L., and Akilov, G., 1964, Functional Analysis in Normed Spaces. Translated by D. Brown, and A. Robertson, Pergamon Press, Oxford.

63. Keifer, J., 1953, Sequential Minimax Search for a Maximum, Proc. Amer. Math. Soc. 34, 503-506.

64. Kowalik, J., and Osborne, M., 1968, Methods for Unconstrained Optimization Programs. American Elsevier, New York.

65. Lawson, C.L., and Hanson, R.J., 1974, Solving Least Squares Problems, Prentice-Hall Inc.

66. Levenberge, K., 1944, A Method for the Solution of Certain Nonlinear Problems in Least Squares, Quart. J. Appl. Math. 2, 164-168.

67. Madsen, K., 1973, An Algorithm for Minimax Solution of Overdetermined Systems of Nonlinear Equations. AERE Report TP 559.

68. Marquardt, D.W., 1963, An Algorithm for Least Square Estimation of Nonlinear Parameters, SIAM. J. Appl. Math. 11, 431-441.

69. Mathews, A., and Davies, D., 1971, A Comparison of Modified

Newton Methods for Unconstrained Optimization, Comp. J. 14, 293-294.

70. Meyer, R.R., and Roth, P.M., 1972, Modified Damped Least Squares: An Algorithm for Nonlinear Estimation, J. Inst. Math. Appl. 9, 218-233.

71. Moré, J.J., and Trangenstein, J.A., 1976, On the Global Convergence of Broyden's Method, Math. Comp. 30, 523-540.

72. Murray, W., 1972, Second Derivative Methods, in "Numerical Methods for Unconstrained Optimization" (Murray, W. (Ed.)) Academic Press Inc.

73. NAG, Numerical Analysis Group Program Library, IBM 360/370 Mark VI Implementation, NAG Centre Office, 7 Banbury Rd., Oxford OX2 6NN.

74. Ortega, J.M., and Rheinboldt,W.C., 1970, Iterative Solution of Nonlinear Equations in Several Variables. Academic Press Inc.

75. Osborne, M.R., 1972, Some Aspects of Nonlinear Least Squares Calculations in "Numerical Methods for Nonlinear Optimization", (Lootsma, F. (Ed.)), Academic Press, New York and London.

76. Pearson, J.D., 1969, Variable Metric Methods for Minimization. Comp. J. 12, 171-178.

77. Powell, M.J.D., 1962, An Iterative Method for Finding Stationary Values of a Function of Several Variables. Comp. J. 5, 147-151.

78. Powell, M.J.D., 1971, Recent Advances in Unconstrained optimization, Math. Prog. 1, 26-57.

79. Powell, M.J.D., 1971a, On the Convergence of the Variable Metric Algorithms, J. Inst. Math. Appl. 7, 21-36.

80. Powell, M.J.D., 1975, Convergence Properties of a Class of Minimization Algorithms, in "Nonlinear Programming 2" O.L. Mangassarian, R.R. Meyer, and S.M. Robinson (Ed.).

81. Powell, M.J.D., 1976, Some Global Convergence Properties of a Variable Metric Algorithm for Minimization Without Exact Line Searches, in "Nonlinear Programming, SIAM - AMS proceeding 9, Amer. Math. Soc. Providence, R.I."

82. Preyera, V., 1967, Iterative Methods for Solving Nonlinear Least Squares Problems, SIAM J. Numer. Anal. 4, 27-36.

83. Rall, L., 1969, Computational Solution of Nonlinear Operator Equations. Wiley, New York.

84. Rosenbrock, H.H., 1960, An Automatic Method for Finding the Greatest or Least Value of a Function. Comp. J. 3, 175-184.

85. Shanno, D.F., 1970, Conditioning of Quasi-Newton Methods for Function Minimization, Math. Comp. 24, 647-656.

86. Simmons, D.M., 1975, Nonlinear Programming for Operations Research, Prentice-Hall International Series in Management.

87. Stoer, J., 1975, On the Convergence Rate of Improved Minimization Algorithms in Broyden's $\beta$ - Class, Math. Prog. 9, 315-335.

88. Traub, J., 1964, Iterative Methods for the Solution of Equations, Prentice-Hall, Englewood Cliffs, New Jersey.

89. Wolfe, M.A., 1976, Some Methods for Least Squares Estimation, J. Inst. Math. Appl. 18, 219-236.

90. Wolfe, M.A., 1978, Numerical Methods for Unconstrained Optimization an Introduction, Van Nostrand Reinhold.

91. Wolfe, M.A., 1978a, Extended Iterative Methods for the Solution of Operator Equations. Numer. Math. 31, pp 153-174.

92. Wolfe, M.A., 1978b, Private Communication.

93. Zangwill, W.J., 1967, Nonlinear Programming, Via Penalty Functions. Man. Sci. 13, 344-358.

## Appendix 1

### Test Problems Used in Sections 6.2 and 6.3.

Problem 1.1. Rosenbrock (1960).

$$F(x) = 100\,(x_1^2 - x_2)^2 + (1 - x_1)^2.$$

$$x^{(0)} = (-1.2,\ 1)^T.$$

Problem 1.2. Wood (see Pearson (1969) ).

$$F(x) = 100\,(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90\,(x_3^2 - x_4)$$

$$+ 10.1\left\{(x_2 - 1)^2 + (x_4 - 1)^2\right\} + 19.8\,(x_2 - 1)(x_4 - 1).$$

$$x^{(0)} = (-3,\ -1,\ -3,\ -1)^T.$$

Problem 1.3. Miele and Cantrell (1969).

$$F(x) = (\exp(x_1) - x_2)^4 + 100\,(x_2 - x_3)^6 + (\tan^{-1}(x_3 - x_4))^4 + x_1^8$$

$$x^{(0)} = (1,\ 2,\ 2,\ 2)^T, \qquad m = 3.$$

Problem 1.4. Powell (1962).

$$F(x) = (x_1 + 10x_2)^2 + 5\,(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10\,(x_1 - x_4)^4.$$

$$x^{(0)} = (3,\ -1,\ 0,\ 1)^T.$$

Problem 1.5. Fletcher and Powell (1963).

$$F(x) = 100\left( (x_3 - 10\theta)^2 + (r-1)^2 \right) + x_3^2 ,$$

where

$$r = \left| (x_1^2 + x_2^2) \right| , \text{ and}$$

$$\theta = \begin{cases} \frac{1}{2\pi} \tan^{-1}(x_2/x_1) & (x_1 > 0) \\ \frac{1}{2\pi} \tan^{-1}(x_2/x_1) + \frac{1}{2} & (x_1 < 0). \end{cases}$$

$$x^{(0)} = (-1.0,\ 0.0,\ 0.0)^T.$$

Problem 1.6 Box (1966).

$$F(x) = \sum_{i=1}^{10} \left[ (\exp(kx_1/10) - \exp(-kx_2/10)) - (\exp(-k/10) - \exp(-k)) \right]^2.$$

$$x^{(0)} = (5,\ 0)^T .$$

Problem 1.7 Biggs (1971).

$$F(x) = \sum_{i=1}^{10} \left[ (\exp(-kx_1/10) - 5\exp(-kx_2/10)) - (\exp(-k/10) - 5\exp(-k)) \right]^2.$$

$$x^{(0)} = (1,\ 2)^T .$$

Problem 1.8 Biggs (1971).

$$F(x) = \sum_{i=1}^{10} \left[ (\exp(-kx_1/10) - x_3\exp(-kx_2/10) - (\exp(-k/10) - 5\exp(-k)) \right]^2.$$

$$x^{(0)} = (1,\ 2,\ 1)^T.$$

Problem 1.9. Biggs (1971).

$$F(x) = \sum_{i=1}^{10} \left[ (x_3 \exp(-kx_1/10) - x_4 \exp(-kx_2/10)) - (\exp(-k/10) - 5 \exp(-k)) \right]^2.$$

$$x^{(0)} = (1,\ 2,\ 1,\ 1)^T.$$

Problem 1.10 Dixon (1973a).

$$F(x) = (1 - x_1)^2 + (1 - x_{10})^2 + \sum_{i=1}^{9} (x_i^2 - x_{i+1})^2.$$

$$x^{(0)} = (-2,\ \ldots,\ -2)^T,\quad m = 3.$$

Problem 1.11.

$$F(x) = (1 - x_1)^2 + (1 - x_6)^2 + \sum_{i=1}^{5} (x_i^2 - x_{i+1})^2.$$

$$x^{(0)} = (-2,\ \ldots,\ -2)^T,\quad m = 3.$$

Problem 1.12.

$$F(x) = (1 - x_1)^2 + (1 - x_8)^2 + \sum_{i=1}^{7} (x_i^2 - x_{i+1})^2.$$

$$x^{(0)} = (-2,\ \ldots,\ -2)^T,\quad m = 3.$$

Problem 1.13.

$$F(x) = x_1^8 + (\exp(x_1) - x_2)^4 + 100(x_2 - x_3)^6 + \left[\tan^{-1}(x_3 - x_4)\right]^4 + (1 - x_5)^2 + (1 - x_6)^2 + (x_4^2 - x_5)^2 + (x_5^2 - x_6)^2.$$

$$x^{(0)} = (1, 2, 2, 2, -2, -2)^T, \quad m = 3.$$

Problem 1.14.

$$F(x) = (1 - x_1)^2 + (1 - x_{10})^2 + \sum_{i=1}^{8}\left((x_i^2 - x_{i+1})^2 + (x_i^2 - x_{i+1})^2\right) + (x_9^2 - x_{10})^2.$$

$$x^{(0)} = (-2, \ldots, -2)^T, \quad m = 5.$$

Problem 1.15.

$$F(x) = (1 - x_1)^2 + (1 - x_{10})^2 + (x_8^2 - x_9)^2 + (x_8^2 - x_{10})^2 + (x_9^2 - x_{10})^2 + \sum_{i=1}^{7}\left((x_i^2 - x_{i+1})^2 + (x_i^2 - x_{i+2})^2 + (x_i^2 - x_{i+3})^2\right).$$

$$x^{(0)} = (-2, -2, \ldots, -2)^T, \quad m = 7.$$

Problem 1.16 Biggs (1971).

$$F(x) = \sum_{i=1}^{13} \left( x_3 \exp(-x_1 t_i) - x_4 \exp(-x_2 t_i) + x_6 \exp(-x_5 t_i) - y_i \right)^2 ,$$

where

$$y_i = \exp(-t_i) - 5\exp(-10t_i) + \exp(-4t_i) \quad (i = 1, \ldots, 13),$$

in which $t_i = i/10$ $(1 \leqslant i \leqslant 13)$.

$$x^{(0)} = (1, 2, 1, 1, 1, 1)^T .$$

Problem 1.17 Brent (1971).

$$F(x) = x_1^2 + (x_2 - x_1^2 - 1)^2 + \sum_{i=2}^{30} \left\{ \sum_{j=2}^{n} (j-1) \left[ (i-1)/29 \right]^{j-2} - \left( \sum_{j=1}^{n} x_j \left[ (i-1)/29 \right]^{j-1} \right)^2 - 1 \right.$$

$$x^{(0)} = (0, 0, \ldots, 0)^T , \quad n = 9.$$

Problem 1.18. Brent (1971).

F is as in Problem 1.17 with $n = 6$.

Problem 1.19. Gill, Murray, and Pitfield (1972).

$$F(x) = a \sum_{i=1}^{n} (x_i - 1)^2 + b \left\{ \sum_{i=1}^{n} x_i^2 - 1/4 \right\}^2 , \text{ where}$$

$$a = 10^{-5} , \quad b = 1.$$

$$x^{(0)} = (1, 2, \ldots, n)^T , \quad n = 10.$$

Problem 1.20. Gill, Murray, and Pitfield (1972).

F is as in Problem 1.19 with $a = 10^{-3}$.

Problem 1.21. Gill, Murray, and Pitfield (1972).

$$F(x) = a\left[\sum_{i=2}^{n}\left(\exp(x_i/10) + \exp(x_{i-1}/10) - C_i\right)^2 + \right.$$

$$\left. (\exp(x_1/10) - \exp(-1/10))^2\right]$$

$$+ b\left\{\left(\sum_{i=1}^{n}(n-i+1)x_i^2\right) + (x_1 - 1/5)^2\right\},$$

where

$$C_i = \exp(x_i/10) + \exp((i-1)/10) \quad (i = 2, \ldots, n),$$

$a = 10^{-5}$, and $b = 1$.

$x^{(0)} = (1/2, 1/2, \ldots, 1/2)^T$, $n = 4$.

Problem 1.22. Gill, Murray and Pitfield (1972).

F is as in Problem 1.21 with $a = 10^{-3}$.

Problem 1.23. Wolfe (1976a).

$$F(x) = (-3x_1 + x_1^2/2 + 2x_1 - 1)^2 + \sum_{i=2}^{9}(x_{i+1} - 3x_i + x_i^2/2 + 2x_{i+1} - 1)^2$$

$$+ (x_9 - 3x_{10} + x_{10}^2/2 - 1)^2.$$

$x^{(0)} = (-1, \ldots, -1)^T$, $m = 5$.

Problem 1.24. Wolfe (1978a).

$$F(x) = (-3x_1 + x_1^2/10 + 2x_2 - 1)^2 + \sum_{i=2}^{4} (x_{i-1} - 3x_i + x_i^2/10 + 2x_{i+1} - 1)^2$$

$$+ (x_4 - 3x_5 + x_5^2/10 - 1)^2 .$$

$$x^{(0)} = (-1, -1, \ldots, -1)^T , \quad m = 5.$$

Problem 1.25.

$$F(x) = 100 \quad (x_3 - 10\theta)^2 + (r - 1)^2 + (x_2 + 10x_4)^2 + 5(x_3 - x_4)^2$$

$$+ (x_3 - 2x_5)^4 + 10(x_4 - x_5)^4 + x_5^2 ,$$

where r and $\theta$ are the same as in Problem 1.3.

$$x^{(0)} = (-1, 0, 3, -1, 0)^T , \quad m = 5.$$

Problem 1.26.

$$F(x) = (1 - x_1)^2 + (1 - x_n)^2 + \sum_{i=1}^{n-1} (x_i^{2p} - x_{i+1})^2 .$$

$$x^{(0)} = (-2, \ldots, -2)^T , \quad n = 4, \; p = 3, \; m = 3.$$

Problem 1.27.

F is as in Problem 1.26 with n = 6.

Problem 1.28.

F is as in Problem 1.26, with n = 10, and p = 5,

Note: the values of m in problems 1.3, 1.10, 1.11 - 1.15 and 1.23 - 1.28 show that the Hessian corresponding to the function is m - diagonal.

## Appendix 2.

Test Problems Used with the Least Squares Algorithms.

Problem 2.1 F is as in Problem 1.1.

Problem 2.2. F is as in Problem 1.5.

Problem 2.3. F is as in Problem 1.4.

Problem 2.4. F is as in Problem 1.2.

Problem 2.5. Zangwill (1967).

$$F(x) = (x_1 - x_2 + x_3)^2 + (-x_1 + x_2 + x_3)^2 + (x_1 + x_2 - x_3)^2,$$

$$x^{(0)} = (100, -1, 2.5)^T.$$

Problem 2.6. Engwall (1966).

$$F(x) = \sum_{i=1}^{5} f_i(x)^2 ,$$

where

$$f_1(x) = x_1^2 + x_2^2 + x_3^2 - 1,$$

$$f_2(x) = x_1^2 + x_2^2 + (x_3 - 2)^2 - 1,$$

$$f_3(x) = x_1 + x_2 + x_3 - 1,$$

$$f_4(x) = x_1 + x_2 - x_3 + 1,$$

$$f_5(x) = x_1^3 + 3x_2^3 + (5x_3 - x_1 + 1)^2 - 36 ,$$

$$x^{(0)} = (1, 2, 0)^T .$$

Problem 2.7. Branin (1971).

$$F(x) = \left[4 (x_1 + x_2)\right]^2 +$$

$$\left[4 (x_1 + x_2) + (x_1 - x_2)(x_1 - 2) + x_2^2 - 1\right]^2 ,$$

$$x^{(0)} = (2, 0)^T .$$

Problem 2.8. Beale (1958), Betts (1976).

$$F(x) = \sum_{i=1}^{3} \left[c_i - x_1 (1 - x_2^i)\right]^2 ,$$

where $c_1 = 1.5$, $c_2 = 2.25$, $c_3 = 2.625$.

$$x^{(0)} = (0.1, 0.1)^T .$$

Problem 2.9. F is as in Problem 1.3.

Problem 2.10 Box (1966), Betts (1976).

$$F(x) = \sum_{i=1}^{10} \left[(\exp(-x_1 t_i) - \exp(-x_2 t_i)) - x_3 (\exp(-t_i) - \exp(-10 t_i))\right]^2 ,$$

where $t_i = 0.1i, \quad (i = 1, 2, \ldots, 10).$

$$x^{(0)} = (0, 10, 20)^T .$$

Problem 2.11. Davidon (1976).

$$F(x) = \sum_{i=1}^{m-1} \left[ \sum_{j=1}^{n} i^{j-1} x \right]^2 + (x_1 - 1)^2 ,$$

where $m = 15, \quad n = 5.$

$$x^{(0)} = (0, 0, 0, 0, 0)^T .$$

Problem 2.12. Freudenstein and Roth (1963).

$$F(x) = f_1(x)^2 + f_2(x)^2 ,$$

where

$$f_1(x) = -13 + x_1 - 2x_2 + 5x_2^2 - x_2^3 ,$$

$$f_2(x) = -29 + x_1 - 14x_2 + x_2^2 + x_2^3 .$$

$$x^{(0)} = (15, -2)^T .$$

Problem 2.13. Watson (see Brent (1971)).

$$F(x) = x_1^2 + (x_2 - x_1^2 - 1)^2 + \sum_{i=2}^{30} (S_{i1} - S_{i2}^2 - 1)^2 ,$$

where $S_{i1} = \sum_{j=2}^{n} (j - 1) x_j ((i - 1)/29)^{j-2}$ and

$$S_{i2} = \sum_{j=1}^{n} x_j \left((i - 1)/29\right) .$$

$$m = 31, \quad n = 6, \quad x^{(0)} = (0, 0, 0, 0, 0, 0)^T .$$

Problem 2.14. Davidon (1976).

$$F(x) = \sum_{i=1}^{m} \left[ \left(x_1 + x_2 t_i - \exp(t_i)\right)^2 + \left(x_3 + x_4 \operatorname{Sin}(t_i) - \cos(t_i)\right)^2 \right]^2 ,$$

where $t_i = 0.2i, \quad m = 20, \quad n = 4.$

$$x^{(0)} = (25, 5, -5, -1)^T .$$

Problem 2.15. Bard (1970).

$$F(x) = \sum_{i=1}^{15} \left[ y_i - \left(x_1 + u_i / \left(x_2 v_i + x_3 w_i\right)\right) \right]^2 ,$$

$u_i = i, \quad v_i = 16 - i, \quad w_i = \min(u_i, w_i)$ and $y_i$ is given in Table 1.

$$x^{(0)} = (1, 1, 1)^T .$$

Problem 2.16. Jennrich and Sampson (1968).

$$F(x) = \sum_{i=1}^{10} \left[ y_i - \left(\exp(-ix_1) + \exp(ix_2)\right) \right]^2 ,$$

where

$$y_i = 2 + 2i, \quad i = 1, 2, \ldots, 10.$$

$$x^{(0)} = (0.3,\ 0.4)^T .$$

Problem 2.17. Kowalik and Osborne (1968).

$$F(x) = \sum_{i=1}^{11} \left[ y_i - (x_1^2 (u_i + x_2 u_i)/(u_i^2 + x_3 u_i + x_4) \right]^2 ,$$

where $y_i$ and $u_i$ are given in Table 2.

$$x^{(0)} = (0.25,\ 0.39,\ 0.415,\ 0.39)^T .$$

Problem 2.18. Osborne (1972).

$$F(x) = \sum_{i=1}^{33} \left[ y_i - (x_1 + x_2 \exp(-x_4 t_i) + x_3 \exp(-x_5 t_i)) \right]^2 ,$$

where $t_i = 10\,(i - 1)$, $i = 1, 2, \ldots, 33$ and $y_i$ is given in Table 3.

$$x^{(0)} = (0.5,\ 1.5,\ -1,\ 0.01,\ 0.02)^T .$$

Problem 2.19. Osborne (1972).

$$F(x) = \sum_{i=1}^{65} (y - A)^2 \quad , \text{ where}$$

$$A = x_1 \exp(-x_5 t_i) + x_2 \exp(-x_6 (t_i - x_9)^2)$$

$$+ x_3 \exp(-x_7 (t_i - x_{10})^2) + x_4 \exp(-x_8 (t_i - x_{11})^2),$$

in which $t_i = 0.1\,(i - 1)$ $i = 1, 2, \ldots, 65$ and

$y_i$ is given in Table 4.

$$x^{(0)} = (1.3,\ 0.65,\ 0.65,\ 0.7,\ 0.6,\ 3,\ 5,\ 7,\ 2,\ 4.5,\ 5.5)^T .$$

Problem 2.20. Madsen (1973).

$$F(x) = (x_1^2 + x_2^2 + x_1 x_2)^2 + \mathrm{Sin}^2 x_1 + \cos^2 x_2 .$$

$$x^{(0)} = (3,\ 1)^T .$$

Problem 2.21 F is as in Problem 2.13 with $n = 9$ .

Problem 2.22. Hartley (1961).

$$F(x) = \sum_{i=1}^{6} (x_1 + x_2 \exp(x_3 t_i) - y_i)^2 ,$$

where $t_i$ , $y_i$ are given in Table 5.

$$x^{(0)} = (580,\ -180,\ -0.16)^T .$$

Problem 2.23. Meyer and Roth (1972).

F is as in Problem 1.1 with

$$x^{(0)} = (-0.86,\ 1.14)^T .$$

Problem 2.24. Meyer and Roth (1972).

$$F(x) = \sum_{i=1}^{5} \left[ x_1 \, x_2 \, u_i / (1 + x_1 \, u_i + x_2 \, v_i) - y_i \right]^2 ,$$

where $u_i$, $v_i$, and $y_i$ $(i = 1, 2, \ldots, 6)$ are given in Table 6.

$$x^{(0)} = (10.39, 48.83, 0.74)^T .$$

Problem 2.25. Pereyra (1967).

$$F(x) = \sum_{i=1}^{13} \left[ x_2 \, \mathrm{Sin}(x_1 \, t_i) + x_3 - y_i \right]^2 ,$$

where

$y_i = \mathrm{Sin}(t_i)$ $(i = 1, 2, \ldots, 13)$ in which $t_i$ are given by the following.

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $t_i$ | 0.105 | 0.25 | 0.4 | 0.55 | 0.7 | 0.9 | 1.1 | 1.25 | 1.35 | 1.45 | 1.55 | 1.57 | 1. |

Pereyra used the values of $y_i$ correct to 3D.

$$x^{(0)} = (0.9, 0.9, 0.1)^T .$$

Problem 2.26 Pereyra (1967).

$$F(x) = \sum_{i=1}^{10} \left[ x_2 \exp(t_i \, x_1) + x_3 - y_i \right]^2 ,$$

where $y_i = 0.1 \exp(t_i) - 5.0$ $(i = 1, 2, \ldots, 10)$ in which

$$t_i = -2.0\ (0.5)\ 2.5.$$

$$x^{(0)} = (0.8,\ 0.2,\ -4.5)^T.$$

Problem 2.27. Meyer and Roth (1972).

$$F(x) = \sum_{i=1}^{10} \left[ x_1 + x_2 \exp(t_i\, x_3) - y_i \right]^2,$$

where the data are generated correct to 4D with

$x = (15.5,\ 1,\ 2,\ 0.02)$ , and are given by Meyer and Roth.

$$x^{(0)} = (20,\ 2,\ 0.5)^T.$$

Problem 2.28 Meyer and Roth (1972).

F is as in Problem 2.27 and the data are obtained by rounding the data of Problem 2.27 to 1D.

$$x^{(0)} = (20,\ 2,\ 0.5)^T.$$

Problem 2.29. Meyer and Roth (1972).

$$F(x) = \sum_{i=1}^{16} \left[ x_1 \exp(x_2 / (t_i + x_3)) - y_i \right]^2,$$

where the data are given by Meyer and Roth.

$$x^{(0)} = (0.02, 4000, 250).$$

Problem 2.30. Meyer and Roth (1972).

$$F(x) = \sum_{i=1}^{23} \left[ x_3 (\exp(-x_1 u_i) + \exp(-x_2 v_i)) - y_i \right]^2$$

where the data are generated with $x = (31.5, 1.5, 20.1)^T$ and are given by Meyer and Roth.

$$x^{(0)} = (12, 1, 25)^T .$$

Problem 2.31 Meyer and Roth (1972).

F is as in Problem 2.30 and the data are given by Meyer and Roth.

Problem 2.32.

F is as in Problem 2.13 with n = 11.

TABLE 1

| i | $y_i$ |
|---|---|
| 1 | 0.14 |
| 2 | 0.18 |
| 3 | 0.22 |
| 4 | 0.25 |
| 5 | 0.29 |
| 6 | 0.32 |
| 7 | 0.35 |
| 8 | 0.39 |
| 9 | 0.37 |
| 10 | 0.58 |
| 11 | 0.73 |
| 12 | 0.96 |
| 13 | 1.34 |
| 14 | 2.10 |
| 15 | 4.39 |

TABLE 2

| i | $y_i$ | $u_i$ |
|---|---|---|
| 1 | 0.1957 | 4.0000 |
| 2 | 0.1947 | 2.0000 |
| 3 | 0.1735 | 1.0000 |
| 4 | 0.1600 | 0.5000 |
| 5 | 0.0844 | 0.2500 |
| 6 | 0.0627 | 0.1670 |
| 7 | 0.0456 | 0.1250 |
| 8 | 0.0342 | 0.1000 |
| 9 | 0.0323 | 0.0833 |
| 10 | 0.0235 | 0.0714 |
| 11 | 0.0246 | 0.0625 |

TABLE 3

| i | $y_i$ | i | $y_i$ |
|---|---|---|---|
| 1 | 0.844 | 18 | 0.558 |
| 2 | 0.908 | 19 | 0.538 |
| 3 | 0.932 | 20 | 0.522 |
| 4 | 0.936 | 21 | 0.506 |
| 5 | 0.925 | 22 | 0.490 |
| 6 | 0.908 | 23 | 0.478 |
| 7 | 0.881 | 24 | 0.467 |
| 8 | 0.850 | 25 | 0.457 |
| 9 | 0.818 | 26 | 0.448 |
| 10 | 0.784 | 27 | 0.438 |
| 11 | 0.751 | 28 | 0.431 |
| 12 | 0.718 | 29 | 0.424 |
| 13 | 0.685 | 30 | 0.420 |
| 14 | 0.658 | 31 | 0.414 |
| 15 | 0.628 | 32 | 0.411 |
| 16 | 0.603 | 33 | 0.406 |
| 17 | 0.580 | | |

TABLE 4

| i | $y_i$ | i | $y_i$ |
|---|---|---|---|
| 1 | 1.366 | 34 | 0.375 |
| 2 | 1.191 | 35 | 0.372 |
| 3 | 1.112 | 36 | 0.391 |
| 4 | 1.013 | 37 | 0.396 |
| 5 | 0.991 | 38 | 0.405 |
| 6 | 0.885 | 39 | 0.428 |
| 7 | 0.831 | 40 | 0.429 |
| 8 | 0.847 | 41 | 0.523 |
| 9 | 0.786 | 42 | 0.562 |
| 10 | 0.725 | 43 | 0.607 |
| 11 | 0.745 | 44 | 0.653 |
| 12 | 0.679 | 45 | 0.672 |
| 13 | 0.608 | 46 | 0.708 |
| 14 | 0.655 | 47 | 0.633 |
| 15 | 0.616 | 48 | 0.668 |
| 16 | 0.606 | 49 | 0.645 |
| 17 | 0.602 | 50 | 0.632 |
| 18 | 0.626 | 51 | 0.591 |
| 19 | 0.651 | 52 | 0.559 |
| 20 | 0.724 | 53 | 0.597 |
| 21 | 0.649 | 54 | 0.625 |
| 22 | 0.649 | 55 | 0.739 |
| 23 | 0.694 | 56 | 0.710 |
| 24 | 0.644 | 57 | 0.729 |
| 25 | 0.624 | 58 | 0.720 |
| 26 | 0.661 | 59 | 0.636 |
| 27 | 0.612 | 60 | 0.581 |
| 28 | 0.558 | 61 | 0.428 |
| 29 | 0.533 | 62 | 0.292 |
| 30 | 0.495 | 63 | 0.162 |
| 31 | 0.500 | 64 | 0.098 |
| 32 | 0.423 | 65 | 0.054 |
| 33 | 0.395 | | |

TABLE 5

| i | $t_i$ | $y_i$ |
|---|---|---|
| 1 | -5 | 127 |
| 2 | -3 | 151 |
| 3 | -1 | 379 |
| 4 | 1 | 421 |
| 5 | 3 | 460 |
| 6 | 5 | 426 |

TABLE 6

| i | $u_i$ | $v_i$ | $y_i$ |
|---|---|---|---|
| 1 | 1.0 | 1.0 | 0.126 |
| 2 | 2.0 | 1.0 | 0.219 |
| 3 | 1.0 | 2.0 | 0.076 |
| 4 | 2.0 | 2.0 | 0.126 |
| 5 | 0.1 | 0.0 | 0.186 |