

RELAXING CONVERGENCE CONDITIONS TO  
IMPROVE THE CONVERGENCE RATE

by

Daniel MacMillan

B.S., University of Colorado, 1976

M.S., Colorado School of Mines, 1985

A thesis submitted to the  
University of Colorado at Denver  
in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
Applied Mathematics

1999

This thesis for the Doctor of Philosophy  
degree by  
Daniel MacMillan  
has been approved  
by

---

Harvey J. Greenberg

---

Richard H. Byrd

---

Gary A. Kochenberger

---

Weldon A. Lodwick

---

Sylvia C.-P. Lu

---

Thomas F. Russell

---

Date

MacMillan, Daniel (Ph.D., Applied Mathematics)

RELAXING CONVERGENCE CONDITIONS TO IMPROVE THE CON-  
VERGENCE RATE

Thesis directed by Professor Harvey J. Greenberg

ABSTRACT

Standard global convergence proofs are examined to determine why some algorithms perform better than other algorithms. We show that relaxing the conditions required to prove global convergence can improve an algorithm's performance. Further analysis indicates that minimizing an estimate of the distance to the minimum relaxes the convergence conditions in such a way as to improve an algorithm's convergence rate.

A new line-search algorithm based on these ideas is presented that does not force a reduction in the objective function at each iteration, yet it allows the objective function to increase during an iteration only if this will result in faster convergence. Unlike the nonmonotone algorithms in the literature, these new functions dynamically adjust to account for changes between the influence of curvature and descent. The result is an optimal algorithm in the sense that an estimate of the distance to the minimum is minimized at each iteration.

The algorithm is shown to be well defined and we prove global convergence of the algorithm. Performance of the algorithm on some standard test functions is presented to illustrate the features of the algorithm.

This abstract accurately represents the content of the candidate's thesis. I recommend its publication.

Signed \_\_\_\_\_  
Harvey J. Greenberg

## ACKNOWLEDGMENTS

I would like to acknowledge Marathon Oil Company for its generous support throughout my pursuit of this degree. I would also like to express my sincerest gratitude to my wife Lois and daughter Michelle for their endless patience and support. Without them this thesis would not have been possible. I am indebted to Dr. H. J. Greenberg whose enthusiasm and knowledge of mathematics in general and optimization in particular have been truly inspiring. I would also like to thank Richard Byrd, Gary Kochenberger, Weldon Lodwick, Sylvia Lu and Thomas Russell for their many valuable comments and help with this thesis. I would also like to acknowledge Allen Holder whose  $\text{\LaTeX}$ template and assistance with  $\text{\LaTeX}$ where invaluable. Finally, I wish to thank my M.S. thesis advisor, Dr. R. G. Underwood, who introduced me to the wonderful world of optimization.

## CONTENTS

<u>Figures</u> . . . . .	viii
<u>Tables</u> . . . . .	ix
<u>Chapter</u>	
1. Introduction . . . . .	1
2. Optimization Preliminaries . . . . .	3
3. Convergence Theory for Optimization . . . . .	11
3.1 Wolfe's Framework . . . . .	11
3.2 Zangwill's Framework . . . . .	15
3.3 Convergence Rate . . . . .	18
4. Relaxing the Conditions for Convergence . . . . .	22
4.1 Relaxing the Sufficient Decrease Condition . . . . .	22
4.2 Relaxing the Bounded Away Condition . . . . .	24
4.3 Relaxing Both Conditions . . . . .	27
5. A New Approach to Algorithm Convergence . . . . .	30
5.1 Examining the Bounded Away Condition . . . . .	30
5.2 Examining the Sufficient Decrease Condition . . . . .	33
5.3 Combining the Results . . . . .	37
6. A Minimum-Distance Function Algorithm . . . . .	40
6.1 Line Search Algorithm . . . . .	47
7. Examples of Algorithm Performance . . . . .	58

7.1	Rosenbrock's Function . . . . .	66
7.2	Freudenstein and Roth's Function . . . . .	76
7.3	Powell's Singular Function . . . . .	79
8.	Extensions and Future Research . . . . .	81
<u>Appendix</u>		
A.	Standard Test Functions . . . . .	86
<u>References</u> . . . . .		96

## FIGURES

Figure	
2.1	Typical trust-region path. . . . . 6
3.1	Region of acceptable points (W) for Wolfe’s framework. . . . . 15
3.2	Region of acceptable points (Z) for Zangwill’s framework. . . . . 18
4.1	Trust-region algorithm for Rosenbrock’s function. . . . . 27
5.1	Example of when increasing $f(x)$ is beneficial. . . . . 34
5.2	Example of when increasing $f(x)$ is not beneficial. . . . . 37
5.3	Regions defined by the minimum-distance function. . . . . 39
6.1	How $\lambda$ affects the minimum-distance function. . . . . 44
6.2	Comparison of nonmonotone and minimum distance regions. . . . . 45
7.1	Minimum-distance function path for $\lambda_1 = 0.95$ . . . . . 68
7.2	Convergence of $\lambda_k$ for $\lambda_1 = 0.95$ . . . . . 69
7.3	Objective function values for $\lambda_1 = 0.95$ . . . . . 70
7.4	Minimum-distance function path for $\lambda_1 = 0.8$ . . . . . 71
7.5	Convergence of $\lambda_k$ for $\lambda_1 = 0.8$ . . . . . 72
7.6	Objective function values for $\lambda_1 = 0.8$ . . . . . 73
7.7	Minimum-distance function path for $\lambda_1 = 0.5$ . . . . . 74
7.8	Convergence of $\lambda_k$ for $\lambda_1 = 0.5$ . . . . . 75
7.9	Objective function values for $\lambda_1 = 0.5$ . . . . . 76

## TABLES

### Table

7.1	Minimum distance algorithm performance on standard test problems. . . . .	62
7.2	Minimum distance algorithm performance on standard test problems continued. . . . .	63
7.3	More's Marquardt algorithm performance on standard test problems. . . . .	64
7.4	More's Marquardt algorithm performance on standard test problems, continued. . . . .	65

## 1. Introduction

This thesis is the result of research on relaxing the standard convergence conditions in order to improve the convergence rate of optimization algorithms. The standard convergence proofs used to prove an optimization algorithm will converge to a local minimum from an arbitrary starting point are examined. This examination, along with a review of the nonmonotone algorithms in the literature, indicate that relaxing these standard conditions could increase the convergence rate of algorithms in many cases.

Two convergence conditions that are central to this thesis are the “bounded away from orthogonality” condition and the “sufficient decrease” condition. An analysis of these two conditions indicates that minimizing a function different than the objective function relaxes the convergence conditions in such a way as to improve an algorithm’s convergence rate. Unlike the nonmonotone algorithms in the literature, these new functions dynamically adjust to account for changes between the influence of curvature and descent. The result is an optimal algorithm in the sense that an estimate of the distance to the minimum is minimized at each iteration.

A new line-search algorithm based on these ideas is presented that does not require the objective function to decrease at any iteration. This algorithm is shown to be well defined and we prove global convergence of the algorithm. Performance of the algorithm on some standard test functions is presented to illustrate the features of the algorithm.

The thesis is organized as follows. The first chapter gives a general discussion of optimization along with definitions needed later in the text. The second chapter gives a detailed description of two common global convergence proofs which are expanded upon in the next chapter. In chapter 4, nonmonotone algorithms from the literature that relax the convergence conditions are shown to obtain faster convergence. Reasons why relaxing the conditions help convergence are studied in detail in the subsequent chapter, which identifies why relaxing the convergence conditions improves algorithm performance and determines when relaxing these conditions is beneficial. The results of these analyses are combined to obtain a new function to minimize that improves algorithm performance. In Chapter 6, a line-search algorithm based on this function is presented and global convergence to a local minimum is proved. The performance of this algorithm is analyzed in the next chapter where the features of the algorithm are illustrated using standard test functions. Finally, the results are summarized and avenues for future research are presented.

## 2. Optimization Preliminaries

Consider the optimization problem of minimizing a real valued objective function  $f(x)$  with continuous second derivatives where  $x \in \Re^m$ , and  $f$  has a minimum at  $x^*$ . A level set for  $f(x)$  is defined as:

**Definition 2.1** The *level set* for a given function  $f(x)$  and a point  $x_k$  is defined to be

$$L_f(x_k) = \{x : f(x) \leq f(x_k)\} . \quad (2.1)$$

The gradient of this function denoted by  $\nabla f(x)$  is an  $m$ -dimensional vector. The Hessian of this function denoted by  $\nabla^2 f(x)$  is an  $m$  by  $m$  matrix. One method of minimizing  $f(x)$  is to search negative gradient directions successively for the minimum value of  $f$ . It has been shown that this steepest descent algorithm exhibits a linear asymptotic convergence rate [73]. In practice, the convergence rate of this method can be quite slow due to the “face effects” of the eigenspaces associated with the minimum and maximum eigenvalues [1].

In order to obtain a faster convergence rate, other algorithms have been considered. One such algorithm is Newton’s method where the next

iterate is obtained from the previous iterate using the formula

$$\Delta x_k = -\nabla^2 f(x_k)^{-1} \nabla f(x_k) \quad , \quad (2.2)$$

where  $x_{k+1} = x_k + \Delta x_k$  and  $x_k$  is the point reached at iteration  $k$ . Newton's method can be shown to exhibit a quadratic convergence rate on many problems in the neighborhood of a solution provided  $\nabla^2 f(x^*)$  is positive definite [73]. However, the simplified Newton's method presented here will not work on most optimization problems without some modifications. Often, a point that is close enough to the minimum so that Newton's method would yield quadratic convergence may not be available. A robust algorithm needs to be able to converge to a local minimum from any arbitrary starting point. Another problem is that the Hessian may not be positive definite, even at the minimum. If the Hessian is not of full rank, then the inverse will not exist and the algorithm, as presented here, becomes undefined. If the Hessian contains negative eigenvalues, the direction obtained could be an ascent, rather than a descent, direction. Many modifications of Newton's algorithm have been suggested to overcome these difficulties. A line search may be used to ensure that  $f(x_{k+1}) < f(x_k)$  in an attempt to guarantee convergence to a local minimum. The Newton direction is often modified to ensure that it is a descent direction. Yet many of these modifications can prevent the desired asymptotic quadratic

convergence of the algorithm.

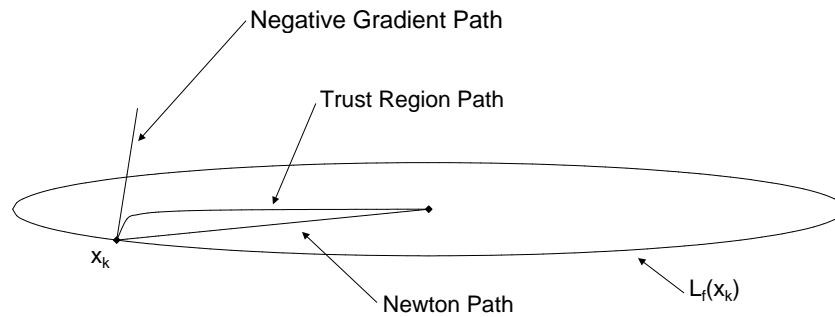
One method of modifying Newton's method that enables it to converge from arbitrary starting points is the model trust region algorithm, where a model of  $f(x)$  near  $x_k$  is obtained. A typical model is the quadratic approximation:

$$m(x) = (x - x_k)^T \nabla^2 f(x_k)(x - x_k) + \nabla f(x_k)^T (x - x_k) + f(x_k) \quad . \quad (2.3)$$

Usually, the model needs to be only a first order approximation of  $f(x)$  to prove global convergence. In a typical trust region algorithm, the model is minimized subject to the trust region constraint  $\|x - x_k\| \leq \Delta_k$ , using an appropriate norm, where  $\Delta_k$  is the current trust region size. A step is then taken using the  $x$  obtained by minimizing the model subject to the trust region constraint. Before the next iteration, the trust region size can be expanded or contracted depending on how well the model predicts the reduction obtained in the objective function  $f(x)$ . An excellent review of trust region algorithms was done by Moré [81]. Shultz et al. [110] obtained a family of trust region algorithms that also account for directions of negative curvature. More recently, Dennis et al. [29] extended the trust region ideas to nonsmooth optimization.

Trust region algorithms have the advantage of yielding a negative gradient step for a small trust region size, ensuring a reduction in  $f(x)$ , while

allowing a full Newton step to be taken when the trust region size is large. This, is achieved by a locus of possible points obtained for various trust region sizes that is nonlinear. An example of a typical trust-region path, illustrated using a quadratic function, is given in Figure 2.1. When a full Newton step results in an increase in  $f(x)$ , smaller trust region sizes are used ensuring that a sufficient reduction in  $f$  will be obtained guaranteeing convergence to a local minimum. When the full Newton step reduces the objective function, the trust region increases allowing the algorithm to take full Newton steps resulting in the desired local quadratic convergence rate.



**Figure 2.1.** Typical trust-region path.

It is often possible to take advantage of special structure that exists in certain types of optimization problems. One example of this is the non-linear least-squares problem:

$$f(x) = \frac{1}{2}R(x)^T R(x) = \frac{1}{2}\|R(x)\|^2, \quad (2.4)$$

where the norm used here is the Euclidean norm with  $R(x) \in \Re^n$ ,  $x \in \Re^m$  and  $m \leq n$ . Let  $r_i(x)$  be the  $i^{\text{th}}$  element of  $R(x)$ . The functions  $r_i(x)$  are often called *residuals*. The gradient of  $f$  is

$$\nabla f(x) = J(x)^T R(x), \quad (2.5)$$

where  $J(x)$  is the Jacobian matrix of first derivatives of  $R(x)$  and we use  $J_i(x)$  to denote the  $i^{\text{th}}$  column of  $J(x)$ . The Hessian of  $f$  is

$$\nabla^2 f(x) = J(x)^T J(x) + \sum_{i=1}^n r_i(x) \nabla^2 r_i(x). \quad (2.6)$$

If  $\nabla^2 f(x)$  is positive definite, Newton's method becomes

$$\Delta x_k = - \left[ J(x_k)^T J(x_k) + \sum_{i=1}^n r_i(x_k) \nabla^2 r_i(x_k) \right]^{-1} J(x_k)^T R(x_k). \quad (2.7)$$

We can take advantage of the special structure of the problem by ignoring the second derivative term

$$S(x) = \sum_{i=1}^n r_i(x) \nabla^2 r_i(x). \quad (2.8)$$

Assuming that  $J(x_k)$  is of full column rank yields the Gauss-Newton method equation

$$\Delta x_k = - [J(x_k)^T J(x_k)]^{-1} J(x_k)^T R(x_k) , \quad (2.9)$$

where  $[J(x_k)^T J(x_k)]^{-1} J(x_k)^T$  is the Moore-Penrose generalized inverse of  $J(x_k)$ . Moré [80] has found a method due to Levenberg [71] and Marquardt [77] to be useful for minimizing the non-linear least-squares problem using a trust region algorithm. The Levenberg-Marquardt algorithm is

$$\Delta x_k = - [J(x_k)^T J(x_k) + \lambda_k I]^{-1} J(x_k)^T R(x_k) , \quad (2.10)$$

where  $\lambda_k > 0$ . Moré solves the trust region problem by choosing  $\lambda_k$  so that  $\|x - x_k\| \approx \Delta_k$ .

The concept of a generalized matrix inverse [114] is useful in obtaining least-squares solutions for any general matrix  $J$ , even if  $J$  is not of full column rank.

**Definition 2.2** The *Moore-Penrose generalized inverse* of an  $n$  by  $m$  matrix  $J$  is defined to be the unique  $m$  by  $n$  matrix  $J^+$  satisfying the following four properties:

$$J^+ J J^+ = J^+ \quad (2.11)$$

$$J J^+ J = J \quad (2.12)$$

$$(JJ^+)^T = JJ^+ \quad (2.13)$$

$$(J^+J)^T = J^+J \quad (2.14)$$

We also have the relations

$$(J^+)^+ = J \quad (2.15)$$

$$(J^T)^+ = (J^+)^T \quad (2.16)$$

$$(JJ^T)^+ = (J^+)^T J^+ \quad (2.17)$$

In the sequel, the following definitions for projection matrices will be found useful

$$P_J = JJ^+ \quad (2.18)$$

$$R_J = J^+J \quad (2.19)$$

Using the Moore-Penrose generalized inverse, the Gauss-Newton method becomes

$$\Delta x_k = -J(x_k)^+ R(x_k) \quad (2.20)$$

This definition makes the Gauss-Newton method well defined, even when  $J(x_k)$  is not of full column rank.

The Moore-Penrose generalized inverse can easily be obtained using the singular value decomposition

$$J = USV^T, \quad (2.21)$$

where  $U$  is an  $n$  by  $n$  unitary orthogonal matrix,  $V$  is an  $m$  by  $m$  unitary orthogonal matrix and  $S$  is an  $n$  by  $m$  matrix with the positive singular values,  $s_i$ , along the diagonal. The Moore-Penrose inverse of  $J$  is obtained with the singular value decomposition using

$$J^+ = VS^+U^T \quad , \quad (2.22)$$

where the pseudo inverse of  $S$ ,  $S^+$ , is an  $m$  by  $n$  matrix with  $1/s_i$  on the diagonal. If  $J$  is not of full column rank, then one or more of the singular values are zero. In this case,  $S^+$  will have a zero on the diagonal whenever  $s_i$  is zero, and  $1/s_i$  otherwise. We will find the singular value decomposition to be a useful analysis tool in the sequel.

### 3. Convergence Theory for Optimization

The above discussion identified two of the numerous major considerations in non-linear optimization algorithm design. One consideration, global convergence, is ensuring that the algorithm converges to a local minimum, given an arbitrary starting point. The other major consideration is obtaining a fast rate of convergence. Two methods of proving that an algorithm is globally convergent have been used extensively. One is a method attributed primarily to Wolfe [124, 125] the other is a method attributed primarily to Zangwill [128]. Convergence rate and these two global convergence proofs are discussed in the following sections.

#### 3.1 Wolfe's Framework

Two independent conditions have been used together for years to prove global convergence of optimization algorithms. The first condition is

$$-\nabla f(x_k)^T d_k \geq c \|\nabla f(x_k)\| \|d_k\| , \quad (3.1)$$

where  $d_k$  is the search direction and  $c$  is some small positive constant. This condition is often referred to as “bounded away from orthogonality” or the

“bounded away condition.” The second condition is

$$f(x_k) - f(x_{k+1}) \geq \sigma \left( \frac{|\nabla f(x_k)^T d_k|}{\|d_k\|} \right), \quad (3.2)$$

where  $\sigma(\cdot)$  is a forcing function. Forcing functions are any non-decreasing function on  $[0, \infty)$  with the property that  $\sigma(t) > 0$  when  $t > 0$  and  $\sigma(t) = 0$  when  $t = 0$ . An example of a forcing function is  $\sigma(t) = \epsilon t^2$ . This condition is often referred to as “sufficient decrease” or the “sufficient decrease condition.”

If an algorithm satisfies both of the above conditions, then global convergence of the algorithm can be proven for a G-differentiable function [89].

**Definition 3.1** A function  $f : D \subseteq \mathfrak{R}^n \rightarrow \mathfrak{R}^m$  is *Gateaux* or (*G*-) *differentiable* at an interior point  $x$  of  $D$  if there exists a linear operator  $A : \mathfrak{R}^n \rightarrow \mathfrak{R}^m$  such that, for any  $h \in \mathfrak{R}^n$ ,

$$\lim_{t \rightarrow 0} \frac{1}{t} \|f(x + th) - f(x) - tAh\| = 0. \quad (3.3)$$

We use the following theorem taken from Ortega and Rheinboldt [89] to prove global convergence.

**Theorem 3.2** Suppose that  $f(x) : D \subseteq \mathfrak{R}^m \rightarrow \mathfrak{R}$  is G-differentiable and bounded below on some set  $D_0 \subseteq D$ , the iterates  $\{x_k\}$  remain in  $D_0$ , and the sufficient decrease condition is satisfied for every iteration  $k \geq 0$ . Then

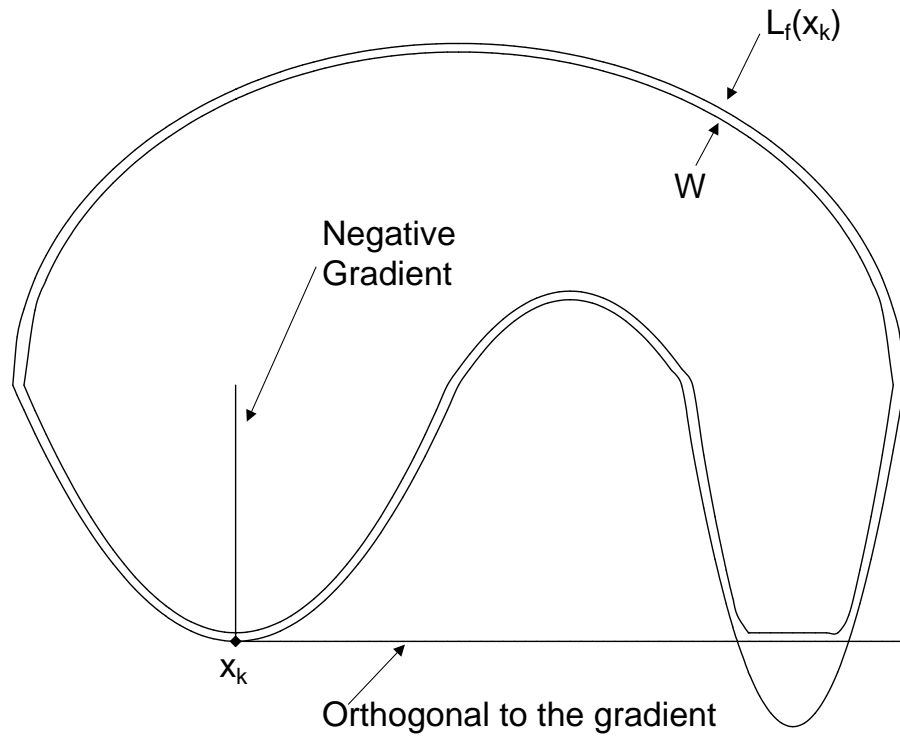
$$\lim_{k \rightarrow \infty} \frac{\nabla f(x_k)^T d_k}{\|d_k\|} = 0. \quad (3.4)$$

This theorem, along with the bounded away from orthogonality condition, ensure the first order convergence result that  $\|\nabla f(x_k)\| \rightarrow 0$ .

To design a practical globally convergent algorithm, one first picks a search direction algorithm that satisfies the bounded away from orthogonality condition. Often, the method of generating the search direction automatically ensures convergence without explicitly enforcing the bounded away from orthogonality condition. Quasi-Newton algorithms and Newton methods using a modified Cholesky decomposition are some examples. Next, a line search method is chosen that satisfies the sufficient decrease condition. Many practical line search algorithms, such as the Goldstein-Armijo method, satisfy this condition [89]. This yields a very powerful theory. A large number of practical globally convergent algorithms are possible by using different search directions with different line search techniques [89].

Let  $W$  denote the region of acceptable points that satisfies these two conditions. The forcing function can be chosen so that only a very small reduction in  $f(x)$  is needed to ensure convergence. The constant  $c$  can be chosen so small that directions nearly perpendicular to the negative gradient direction are acceptable. An illustration of this in two dimensions ( $m = 2$ ) is given in Figure 3.1. The current point is  $x_k$  with objective function value  $f(x_k)$ . Since the function value is required to decrease,  $W \subset L_f(x_k)$  as illustrated in

the figure. The region of acceptable new points is just slightly inside  $L_f(x_k)$ , and it nowhere touches the boundary due to the sufficient decrease condition. There is a small excluded region toward the bottom right where  $L_f(x_k)$  is outside the region defined by the bounded away from orthogonality condition. Points in this region satisfy the sufficient decrease condition but cannot be reached because the search direction required would violate the bounded away from orthogonality condition.



**Figure 3.1.** Region of acceptable points ( $W$ ) for Wolfe's framework.

### 3.2 Zangwill's Framework

Zangwill proposed a global convergence proof based on the idea of point-to-set maps [128]. He defined an algorithm as a point-to-set mapping, where the algorithm uses the current point,  $x_k$ , to generate a set of possible

new iterates. For example, when performing a line search, it is impractical to obtain the exact minimum. Hence, a range of points along the line that are within some tolerance of the minimum are considered acceptable. This range of points becomes the set of points generated by the algorithm in the point-to-set map.

An important property of point-to-set mappings needed to prove global convergence is the idea of a closed mapping [73].

**Definition 3.3** A point-to-set mapping  $A(x)$  from a set  $X$  to subsets of a set  $Y$  is said to be *closed* at  $x \in X$  if the conditions

$$\lim_{k \rightarrow \infty} x_k = x, \quad x_k \in X \tag{3.5}$$

and

$$\lim_{k \rightarrow \infty} y_k = y, \quad y_k \in A(x_k) \tag{3.6}$$

together imply that

$$y \in A(x) \tag{3.7}$$

Using the idea of a point-to-set map and closed mappings, we have the following global convergence theorem [128].

**Theorem 3.4** Let  $A(x)$  be an algorithm on a set  $X$ . Let  $\Gamma \subset X$  be the solution set for the optimization problem. Assume that, given  $x_0$ , the sequence  $\{x_k\}$  is generated satisfying  $x_{k+1} \in A(x_k)$ , and also satisfying the following.

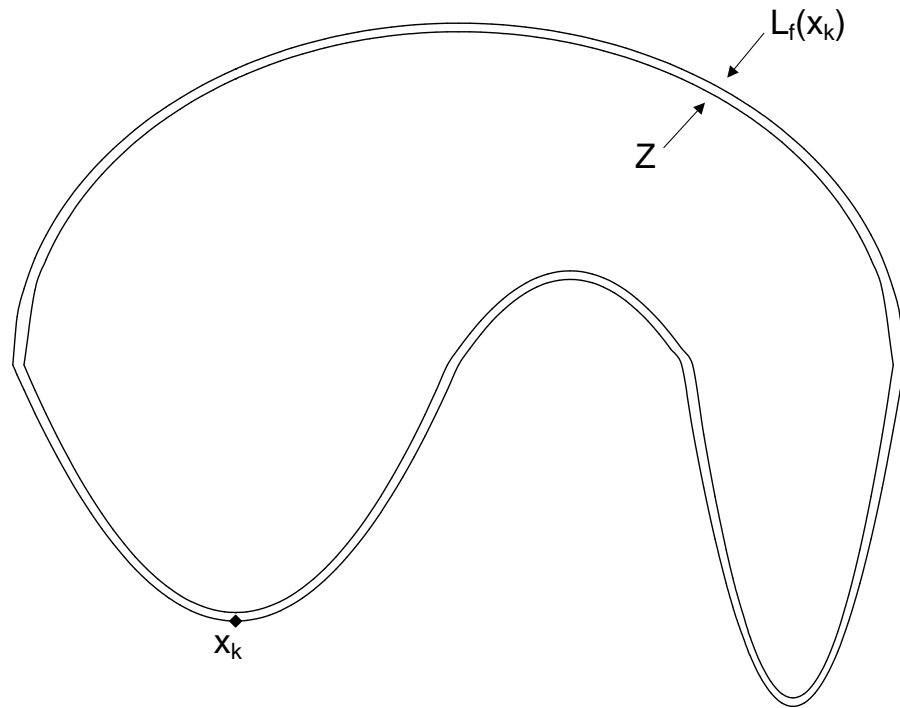
- All  $x_k$  are contained in a compact set  $S \subset X$ .
- There is a continuous function  $g$  on  $X$  such that
  - (1) if  $x \notin \Gamma$  then  $g(y) < g(x) \forall y \in A(x)$
  - (2) if  $x \in \Gamma$  then  $g(y) \leq g(x) \forall y \in A(x)$ .
- The mapping  $A$  is closed at points outside of  $\Gamma$ .

Then, every limit point of  $\{x_k\}$  is a solution.

To use this theorem to prove global convergence for some algorithm, the function  $g$  is usually chosen as the objective function and a solution is usually defined as a stationary point, i.e.  $\nabla f = 0$ . Then, the line search is shown to be a closed point-to-set mapping that reduces the objective function (when not at a solution).

The region  $Z$  of acceptable points that satisfies the conditions of this theorem is again quite large. The conditions on  $g(x) \equiv f(x)$  ensure that  $Z \subset L_f(x_k)$ . An illustration of this for the same two dimensional problem shown in Figure 3.1 is given in Figure 3.2. As can be seen, the region of acceptable new points is just slightly inside  $L_f(x_k)$ , and it nowhere touches the boundary because the mapping is closed and a strict decrease in  $g$  is obtained outside the solution set. Also note that there are no excluded regions like that obtained for Wolfe's proof due to the bounded away from orthogonality condition. However, most practical algorithms will have  $\nabla f(x_k)^T d_k < 0$  for

$\nabla f(x_k) \neq 0$  so that  $f(x)$  can be reduced during the line search. Hence, the two convergence proofs produce essentially the same region of acceptable points.



**Figure 3.2.** Region of acceptable points ( $Z$ ) for Zangwill's framework.

### 3.3 Convergence Rate

Consider the following two definitions commonly used in rate analysis [51].

**Definition 3.5** Let the sequence  $\{x_k\}$  converge to  $x^*$ . The *order* of convergence denoted by  $\alpha$  is defined to be

$$\alpha \equiv \sup\{q : \limsup_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^q} < \infty\} . \quad (3.8)$$

The case  $\alpha = 1$  is called *linear* convergence,  $\alpha = 2$  is called *quadratic* convergence, and  $\alpha = 3$  is called *cubic* convergence.

**Definition 3.6** The convergence *ratio* denoted by  $\beta$  for a sequence  $\{x_k\}$  of order  $\alpha$  is defined to be

$$\beta \equiv \limsup_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^\alpha} . \quad (3.9)$$

The case  $\alpha = 1$  and  $\beta = 0$  is called *superlinear* convergence. These two definitions are useful for algorithm acceleration and the design of convergence criteria. Also, they have often been used as a method of recommending one algorithm over another. However, this use can lead to erroneous conclusions. Larger orders and smaller ratios do not necessarily imply that one algorithm will converge to the solution of a given problem faster [51]. Another problem with using these definitions to compare algorithms is that they are only useful in a region near the solution,  $x^*$ . While this asymptotic convergence rate is important, the analysis of algorithm performance at points far from the solution, where the asymptotic convergence rate does not apply, is also very

important.

There have been many attempts to come up with a good definition for global convergence rate. For steepest descent, the relation

$$f(x_{k+1}) \leq \left[ \frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}} \right]^2 f(x_k) \quad (3.10)$$

has been obtained for a quadratic objective function where  $\lambda_{max}$  and  $\lambda_{min}$  are the largest and smallest eigenvalues of the (constant) Hessian [73]. Others have obtained bounds on  $f(x_k) - f(x^*)$  for certain classes of objective functions. However, an estimate for global convergence rate for more general objective functions remains elusive. A summary of this work is presented in Nocedal [88].

The success obtained in proving global convergence leads us to examine the global convergence proofs given above to see if they contain any insight into algorithm performance. However, such insight is not obtained. Consider an algorithm that always obtains its next point on the boundary of  $W$  or  $Z$ . For Wolfe, this algorithm will always satisfy the sufficient decrease condition with equality. If the forcing function is  $\sigma(t) = \epsilon t^2$ , then the objective function decreases by only  $\epsilon |\nabla f(x_k)^T d_k| / \|d_k\|^2$  at every iteration. This yields a linear convergence rate with a convergence ratio near one for  $\epsilon$  near zero. For Zangwill, the point obtained will be on the boundary of the closed mapping.

Since  $g \equiv f(x)$  is required only to be monotonically decreasing, again very slow convergence is obtained. Thus, we see that the above two global convergence proofs bound the algorithm in question by another algorithm that converges, however, very slowly. This results in a proof of global convergence but yields little insight into algorithm performance at points far from the solution. These global convergence proofs do, however, say something about the reliability of the algorithm. They guarantee that the algorithm is reliable in the sense that it will eventually converge to a local minimum.

## 4. Relaxing the Conditions for Convergence

In the previous chapter, we saw that global convergence theorems do not provide much insight into algorithm performance. Further, because rate analysis is definitionally asymptotic, it is not useful for analyzing the performance of algorithms at points far from the solution. The large regions of acceptance,  $W$  and  $Z$ , illustrated in Figures 3.1 and 3.2, respectively, and the extremely slow convergence of an algorithm limited to the boundary of these regions, indicates that almost any efficient algorithm would fit into the framework provided in the proofs. However, in this chapter we examine evidence that relaxing these convergence conditions can actually lead to algorithms exhibiting faster convergence rates. This evidence is divided into three sections that cover relaxing the sufficient decrease condition, relaxing the bounded away from orthogonality condition and relaxing both the sufficient decrease and bounded away conditions.

### 4.1 Relaxing the Sufficient Decrease Condition

In the mid 1980's, Grippo et al. began a series of papers proposing the advantages of using a nonmonotone line-search method to speed up the

convergence rate. The first paper appeared in 1986 [53], the next in 1989 [54], and another in 1990 [55]. This work was culminated in 1991 with a paper covering an entire class of non-monotone methods [56]. These “non-monotone methods” were summarized, tested and improved upon by Toint, confirming the improved performance of the proposed algorithms [115, 116, 119]. Panier and Tits [91] along with Bonnans et al. [8] immediately saw the advantages of these techniques for constrained optimization problems and applied the nonmonotone line-search idea to help alleviate the Maratos effect in their program FSQP. The Maratos effect arises in constrained optimization, where a reduction in a merit function is often enforced rather than using the objective function because the objective function does not account for the presence of the constraints. Often, the merit function used will not allow a full Newton step to be taken, even at points very close to the solution. This effect, called the Maratos effect [76], prevents Newton’s method from obtaining the asymptotic quadratic convergence rate.

More recently, nonmonotone methods have been applied to a wide variety of algorithms and problems [26, 34, 36, 35, 117, 118, 120, 126, 130]. For other nonmonotone methods see [4] and [74].

These nonmonotone algorithms relax the sufficient decrease condition

by replacing it with the weaker condition

$$\max_{k-p \leq i \leq k} (f(x_i)) - f(x_{k+1}) \geq \sigma \left( \frac{|\nabla f(x_k)^T d_k|}{\|d_k\|} \right) , \quad (4.1)$$

where  $p$  is on the order of 10 or less (the optimization program FSQP currently uses  $p = 3$  [8, 91, 129]). While this condition allows for the relaxing of the sufficient decrease condition, resulting in faster convergence in many cases, it does not explain why relaxing this condition helps. It also does not give a method or algorithm for determining when to relax the sufficient decrease condition and when relaxing the sufficient decrease condition is not a good idea.

## 4.2 Relaxing the Bounded Away Condition

The bounded away from orthogonality condition can be quite restrictive if it is explicitly enforced. For example, consider Newton's method applied to a quadratic problem, where Newton's method converges in just one iteration. However, for a given constant,  $c$ , there exist quadratic problems with an eigenvalue small enough that the full Newton step will violate the bounded away from orthogonality condition. If the Newton step is modified to satisfy the bounded away from orthogonality condition, this will result in destroying the Newton algorithm's performance on this quadratic problem.

Allowing an unmodified Newton step to be taken has proved to be

very computationally efficient in practice. Even though convergence has not been proven for such an algorithm, the inclusion of the bounded away condition has been found to be quite detrimental to the efficient solution of real problems. Hence, relaxing the bounded away from orthogonality condition could, in many cases, improve algorithm performance.

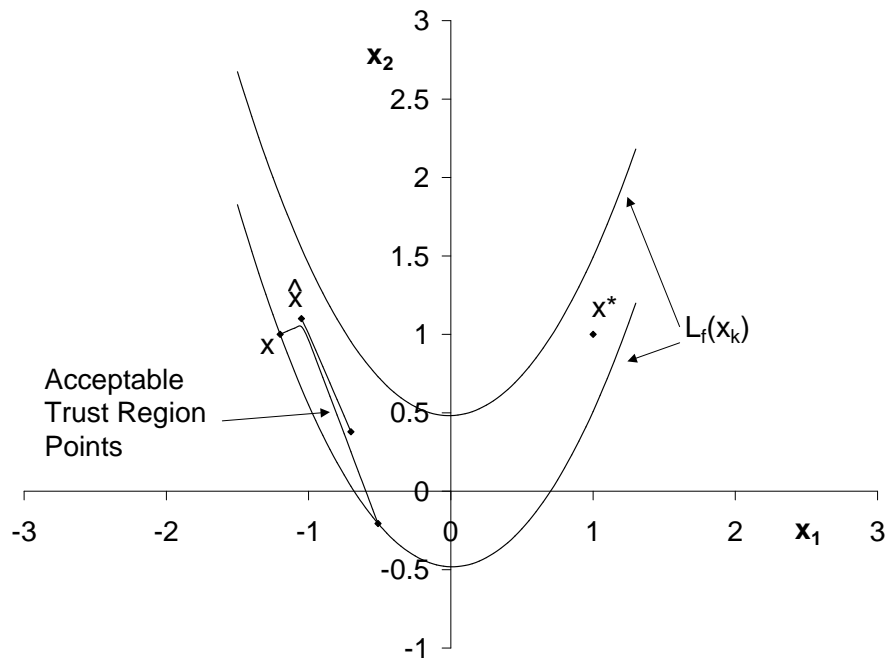
Trust region methods relax the bounded away from orthogonality condition. Rather than checking the step direction to ensure it is a descent direction, trust region methods check how well the model fits the function. At the end of each iteration, the relative function reduction is computed:

$$\rho_k = \frac{f(x_{k+1}) - f(x_k)}{m(x_{k+1}) - m(x_k)} . \quad (4.2)$$

If  $0.75 \leq \rho_k$ , the model fit is considered to be good and the trust region size can be increased. If  $\rho_k \leq 0.25$ , the model fit is not good, and the trust region size is reduced. As the trust region size goes to zero, the step direction approaches the negative gradient direction, so that a sufficient decrease will eventually be obtained ensuring the (theoretical) convergence of the algorithm [81]. In the region where Newton's method is quadratically convergent, the model fit is good, so the trust region size increases, and the model subproblem allows full Newton steps to be taken. Thus, as long as the model is a good approximation to  $f(x)$  near  $x_k$  the step direction need not be a descent direction. However, it

is possible to show that the step to the minimizer of the trust region problem is always a descent direction. Thus, explicit inclusion of the bounded away from orthogonality condition is not needed for trust region algorithms.

However, trust region algorithms could also benefit from relaxing the sufficient decrease condition. Figure 4.1 illustrates one step of a trust region algorithm taken from two different starting points using Rosenbrock's function [102]. One point,  $\hat{x}$ , is near the valley floor while the other point,  $x$ , is farther off the valley floor, yet both points are in close proximity to each other. The locus of acceptable points for a trust region algorithm initiating from both points are also illustrated in the figure. Note that the point near the valley floor,  $\hat{x}$ , has a shorter maximum step length than that obtained for the other point. It would be good if points that are in such close proximity could have the same trust region size. However, if the algorithm were allowed a longer step length from  $\hat{x}$ , then the objective function would have to increase. This illustrates the utility of relaxing the sufficient decrease condition for trust region algorithms in certain cases. Some authors have applied the above nonmonotone ideas to trust region algorithms with some success [26, 117, 118, 120, 126].



**Figure 4.1.** Trust-region algorithm for Rosenbrock’s function.

### 4.3 Relaxing Both Conditions

The above discussion indicates that relaxing both the bounded away and sufficient decrease conditions could be helpful. Here we discuss work that has been done to relax both conditions.

The “watchdog method” due to Chamberlain et al. [14] is an attempt to avoid the Maratos effect in constrained optimization. The basic idea behind

the watchdog method is to take several Newton steps without checking any conditions until after the last step. If the merit function has sufficiently decreased, then the Newton steps will be accepted. If not, the Newton steps are discarded, the algorithm goes back to the point where the Newton steps were started, and the step size is reduced. Thus, this method relaxes both conditions since no conditions are checked during the Newton steps. This method can be quite effective. It can also increase the computer time required, and there is the risk of taking a bad step, exponential overflow or other difficulties. This idea has been incorporated by Grippo et al. [56] in their class of non-monotone methods.

Solodov [75, 111] weakens both conditions by picking a direction  $d_k$  using

$$-\nabla f(x_k)^T d_k \geq \sigma(\|\nabla f(x_k)\|) - \lambda_k \quad , \quad (4.3)$$

where  $\lambda_k \geq 0$ ,  $\sigma(\cdot)$  is a forcing function, and picking a step size  $\eta_k$  so that

$$f(x_k) - f(x_{k+1}) \geq -\eta_k \nabla f(x_k)^T d_k - \nu_k \quad (4.4)$$

with  $\eta_k > 0$  and  $\nu_k \geq 0$ . He proves convergence when the following conditions hold:

$$\sum_{i=0}^{\infty} \eta_k = \infty, \quad \sum_{i=0}^{\infty} \lambda_k \eta_k < \infty, \quad \sum_{i=0}^{\infty} \nu_k < \infty \quad . \quad (4.5)$$

These conditions relax both the original bounded away from orthogonality and

sufficient decrease conditions. Solodov was able to show that some of the algorithms used to train neural networks fit within this structure.

## 5. A New Approach to Algorithm Convergence

In this section we examine the two convergence conditions more closely to determine when relaxing each condition helps speed convergence and when it does not. The results are then combined to yield a new approach to algorithm convergence.

### 5.1 Examining the Bounded Away Condition

The bounded away from orthogonality condition becomes a problem when the search direction denoted by  $d_k \equiv d(x_k)$  is nearly orthogonal to the negative gradient direction. The objective function will not decrease very fast along such a direction, and will likely increase within a short distance of  $x_k$ . This suggests minimizing an alternate function,  $h(x)$ , with the following properties.

(1)  $-\nabla h(x) = d(x)$  so that  $-\nabla h(x_k) = d_k$ .

(2)  $x^*$  minimizes  $h(x)$ .

(3)  $h(x)$  has no local minimum other than  $x^*$ .

A function that satisfies the first condition is called a gradient mapping [89].

**Definition 5.1** A mapping  $d(x) : D \subseteq \mathfrak{R}^m \rightarrow \mathfrak{R}^m$  is a *gradient mapping* on a subset  $D_0 \subseteq D$  if there exists a G-differentiable function  $g : D_0 \subseteq \mathfrak{R}^m \rightarrow \mathfrak{R}^1$  such that  $d(x) = \nabla g(x)$  for all  $x \in D_0$ .

Such a function, if it exists, would solve the bounded away from orthogonality problem because the desired search direction would be the negative gradient direction of the new function being minimized. The nonmonotone idea of allowing  $f(x)$  to increase also becomes less of an issue because  $h(x)$  decreases fastest along its negative gradient direction.

Ortega and Rheinboldt state the following Theorem [89].

**Theorem 5.2** Let  $d(x) : D \subseteq \mathfrak{R}^m \rightarrow \mathfrak{R}^m$  be continuously differentiable on an open convex set  $D_0 \subseteq D$ . Then,  $d(x)$  is a gradient mapping on  $D_0$  if, and only if,  $\nabla d(x)$  is symmetric for all  $x \in D_0$ .

Thus, the desired function exists if, and only if,  $\nabla d(x)$  is symmetric. This condition obviously holds for  $d_k = -\nabla f(x_k)$ . But, this condition is very restrictive so that there are very few search directions that satisfy this condition.

However, there are functions that relax the first condition while satisfying the other two conditions. Consider the function

$$h(x) = \frac{1}{2} \|\nabla^2 f(x)^{-1} \nabla f(x)\|^2, \quad (5.1)$$

where the norm used is the Euclidean norm. The gradient of this function is

$$\begin{aligned} \nabla h(x) = & \left[ \nabla^2 f(x)^{-1} \nabla^2 f(x) \right]^T \nabla^2 f(x)^{-1} \nabla f(x) + \\ & \left[ \sum_{i=1}^m \nabla(\nabla^2 f(x)_i^{-1}) \frac{\partial f(x)}{\partial x_i} \right]^T \nabla^2 f(x)^{-1} \nabla f(x) \quad , \quad (5.2) \end{aligned}$$

where  $\nabla^2 f(x)_i^{-1}$  is the  $i^{\text{th}}$  column of  $\nabla^2 f(x)^{-1}$ . At  $x_k$  we have

$$\begin{aligned} \nabla h(x_k) = & -d_k + \\ & \left[ \sum_{i=1}^m \nabla(\nabla^2 f(x_k)_i^{-1}) \frac{\partial f(x_k)}{\partial x_i} \right]^T \nabla^2 f(x_k)^{-1} \nabla f(x_k) \quad . \quad (5.3) \end{aligned}$$

The second higher order term is the error obtained when using this  $h(x)$  to approximate the first condition. Minimizing this  $h(x)$  is like minimizing the square of an estimate of the distance to the minimum. Hence, we will call all such functions *minimum distance functions*.

This particular minimum distance function has several problems. The derivatives of  $h(x)$  needed for a steepest descent or Newton algorithm are difficult to obtain. The gradient and Hessian of the objective function,  $f(x)$ , are needed at every point in a line search. This function could also find a maximum rather than a minimum. To alleviate some of these problems, one could minimize a sequence of functions

$$h_k(x) = \frac{1}{2} \|\nabla^2 f(x_k)^{-1} \nabla f(x)\|^2 \quad . \quad (5.4)$$

The gradient of this function is

$$\nabla h_k(x) = \left[ \nabla^2 f(x_k)^{-1} \nabla^2 f(x) \right]^T \nabla^2 f(x_k)^{-1} \nabla f(x) \quad . \quad (5.5)$$

At  $x_k$  we now obtain  $-\nabla h(x_k) = d_k$  as desired. However, we must ensure that minimizing a sequence of such functions will converge, and that it will converge to  $x^*$  and not some other point. The idea behind this definition is to have a trust region for how far the current Hessian approximates the eigenvectors and eigenvalues of the problem. Here again, the gradient of the function is needed at each point during a line search.

Now, we extend this idea to least-squares problems where the Gauss-Newton algorithm can be used to take advantage of the special structure of the problem, see page 7. This yields the equation

$$h_k(x) = \frac{1}{2} \|J(x_k)^+ R(x)\|^2 \quad . \quad (5.6)$$

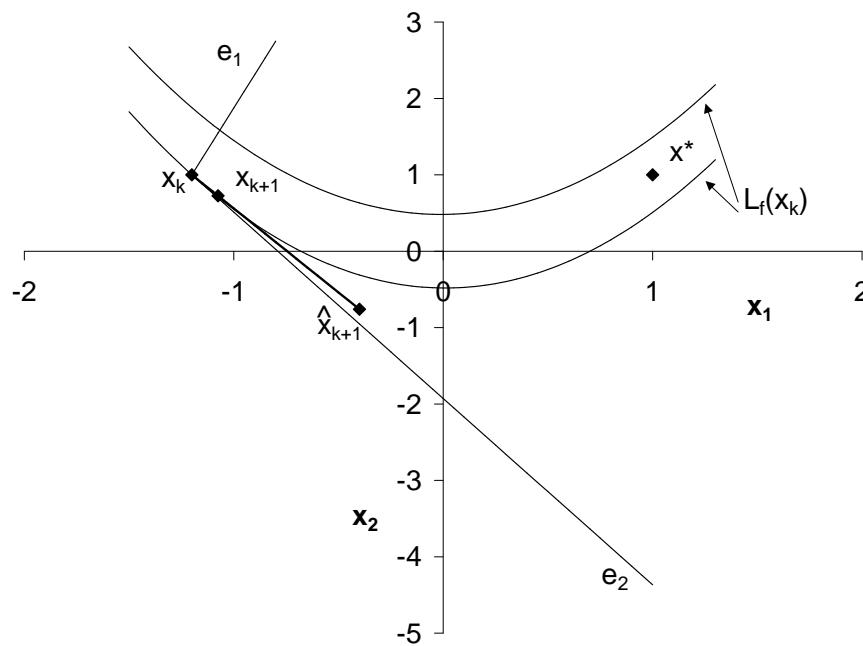
In this case, when  $f(x)$  is evaluated,  $R(x)$  is also obtained because  $f(x) = \frac{1}{2} R(x)^T R(x)$ . Thus, there is no need to evaluate the gradient at each point in the line search. Hence, this minimum distance function could be useful for relaxing the bounded away from orthogonality condition.

## 5.2 Examining the Sufficient Decrease Condition

In this section we address the question, “When is allowing  $f(x)$  to increase a good idea?” The goal is to obtain an algorithm that can automatically

determine the best point along the search direction.

An obvious case where increasing  $f(x)$  along the search direction is beneficial is illustrated in Figure 5.1 using a Rosenbrock type function. Two possible next iterates,  $x_{k+1}$  and  $\hat{x}_{k+1}$  are contrasted in the figure. The point  $x_{k+1}$  is obtained by minimizing the objective function along the typical Newton or Gauss-Newton search direction. The point  $\hat{x}_{k+1}$  is some arbitrary point farther along the search direction, where  $f(\hat{x}_{k+1}) > f(x_k)$ .



**Figure 5.1.** Example of when increasing  $f(x)$  is beneficial.

Note that  $\hat{x}_{k+1}$  is not necessarily the point obtained by taking a full Newton or Gauss-Newton step. This point could be much farther out due to near rank deficiency in the Hessian and may cause an exponential overflow or other numerical problems.

The idea illustrated in Figure 5.1 is that it is easier to minimize  $f(x)$  along eigenvector directions with a large associated eigenvalue ( $e_1$  in the figure) than it is to minimize  $f(x)$  along eigenvector directions with a very small associated eigenvalue ( $e_2$  in the figure) [74]. The objective function is allowed to increase in order to make progress along the eigenvector directions with small associated eigenvalues. Thus, instead of minimizing  $f(x)$ , a new function,  $h_k(x)$ , is minimized that weights the eigenvector directions with small associated eigenvalues more than the eigenvector directions with large associated eigenvalues. Here we include the subscript  $k$  to indicate that the eigenvectors and eigenvalues are obtained at  $x_k$ .

Let us design a new function for nonlinear least-squares objective functions that has this property. Let  $J(x_k) = USV^T$  be the singular value decomposition of the current Jacobian matrix. Also, assume that  $J$  is of full column rank. Next, we rotate the residual vector by forming  $U^T R(x)$ . We now

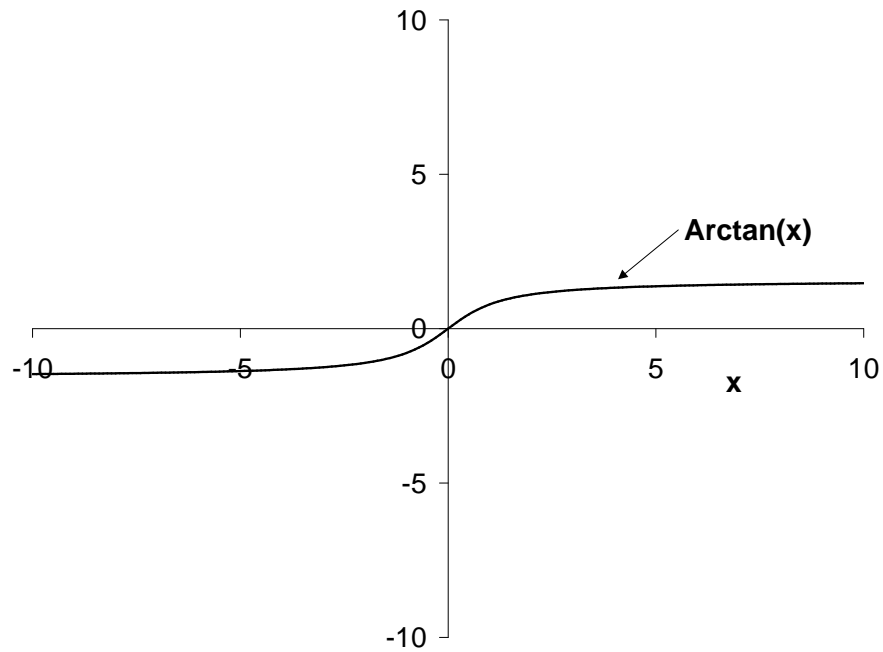
define our new function to be

$$h_k(x) = \frac{1}{2} \|DU^T R(x)\|^2 \quad , \quad (5.7)$$

where  $D$  is a diagonal matrix of scale factors that we shall use to scale the eigenvector directions. Note that if  $D = I$ , then  $h_k(x) = f(x)$ . We now pick the diagonal elements of  $D$  denoted here by  $d_i$  so that they give a small weight to the large  $s_i$  and a large weight to the small  $s_i$ . One easy way to do this is to set  $d_i = 1/s_i$ . This yields exactly the same minimum distance function obtained above for nonlinear least-squares problems because

$$\frac{1}{2} \|DU^T R(x)\|^2 = \frac{1}{2} \|VS^+U^T R(x)\|^2 = \frac{1}{2} \|J(x_k)^+ R(x)\|^2 \quad . \quad (5.8)$$

Now, consider another case illustrated in Figure 5.2, a plot of  $\arctan(x)$ , which is an example of when it is not beneficial to increase  $f(x)$ . Assume that the objective function is  $f(x) = \arctan(x)^2$ , with minimum at the origin. If Newton's method is started at some point far from the origin, it will take increasingly larger steps zig-zagging across the origin yielding ever increasing objective function values until it eventually diverges to  $\pm\infty$ . If this example is minimized using a minimum distance function with constant eigenvectors and eigenvalues, the minimum will still be obtained. Since  $x \in \mathfrak{R}^1$ ,  $J(x_k)^+ = 1/s_1$  is a constant and  $h_k(x) = \frac{1}{2} \|r_1(x)/s_1\|^2$ . Thus,  $h_k(x)$  is just the objective function modified by a constant factor, so that the minimum is not affected.

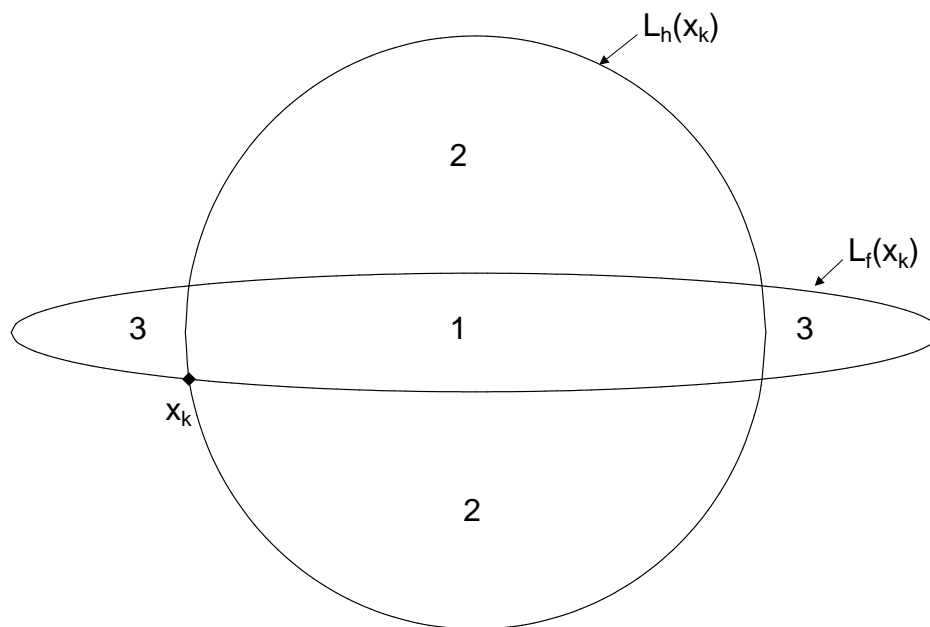


**Figure 5.2.** Example of when increasing  $f(x)$  is not beneficial.

### 5.3 Combining the Results

The above results of examining the two convergence criteria have both amazingly suggested minimizing a minimum distance function along the search direction. It is instructive to examine what this idea does to a quadratic function where,  $L_f(x_k)$  is an ellipse. In this case,  $L_h(x_k)$  will be a circle, as shown in Figure 5.3, because Newton solves a quadratic function exactly. For

the sufficient decrease condition, the new point  $x_{k+1}$  must lie inside  $L_f(x_k)$ , regions 1 and 3 in the figure. If we require  $h(x)$  to decrease along the line-search direction, then the new point  $x_{k+1}$  must lie inside  $L_h(x_k)$ , regions 1 and 2 in the figure. Region 1 is ideal because both the objective function and the Newton step length are decreasing. Region 2 is the region we seek. Points in region 2 have  $f(x_{k+1}) > f(x_k)$  but the Newton step length is decreasing so convergence can still be obtained. Points where  $f(x_{k+1}) > f(x_k)$  that are outside of Region 2 are not acceptable because the Newton step length is increasing. If the algorithm accepts points in this region, convergence may not be obtained. Region 3 is very interesting. Points in this region have a lower objective function value but have an increased distance to the minimum. Requiring  $h_k(x)$  to decrease along the search direction will exclude these points from consideration.



**Figure 5.3.** Regions defined by the minimum-distance function.

If  $h_k(x)$  is a good approximation for the distance to the minimum, minimizing  $h(x)$  makes sense. It allows each step of the algorithm to make the greatest possible progress toward the minimum and is, at least in this sense, optimal.

## 6. A Minimum-Distance Function Algorithm

In this section we consider minimizing a sequence of functions

$$h_k(x) = \frac{1}{2} \|J(x_k)^+ R(x)\|^2 \quad , \quad (6.1)$$

to take advantage of the special structure, see page 7. We often denote  $f(x_k)$  by  $f_k$ ,  $R(x_k)$  by  $R_k$  and  $J(x_k)$  by  $J_k$  to simplify the notation when no confusion will result.

These functions have several nice properties. For example, consider the gradient

$$\nabla h_k(x) = \left( J(x_k)^+ J(x) \right)^T J(x_k)^+ R(x) \quad (6.2)$$

and Hessian

$$\nabla^2 h_k(x) = \left( J(x_k)^+ J(x) \right)^T \left( J(x_k)^+ J(x) \right) + \sum_{i=1}^n y_i(x) \nabla^2 r_i(x) \quad (6.3)$$

where  $y_i(x)$  is the  $i^{\text{th}}$  element of  $Y_k(x)$  defined by:

$$Y_k(x) = \left( J(x_k)^+ \right)^T J(x_k)^+ R(x) \quad . \quad (6.4)$$

Here we see that  $Y_k(x_k) \rightarrow 0$  as  $x_k \rightarrow x^*$  if  $J(x)^+$  is continuous and bounded in a neighborhood of  $x^*$ . It would be nice if this led to superlinear convergence

for the associated Gauss-Newton Method. However, such is not the case. The negative gradient at  $x_k$  is

$$-\nabla h_k(x_k) = -(J(x_k)^+ J(x_k))^T J(x_k)^+ R(x_k) = -J(x_k)^+ R(x_k) . \quad (6.5)$$

Ignoring the higher order terms in the Hessian yields the Gauss-Newton step

$$\Delta x_k = -[(J_k^+ J_k)^T (J_k^+ J_k)]^{-1} (J_k^+ J_k)^T J_k^+ R_k \quad (6.6)$$

$$= -J_k^+ R_k . \quad (6.7)$$

Thus, the negative gradient direction and the Gauss-Newton direction for  $h_k(x)$  coincide. These directions coincide because the first term in the Hessian is  $R_J^T R_J = R_J$ , a unitary orthogonal matrix, that projects onto the row space of  $J(x_k)$ . This gives the Hessian of  $h_k(x)$  a much better condition number than the Hessian of  $f(x)$  when the higher order terms,  $S(x)$  on page 7, are small compared to the first order terms. In this case,  $\nabla^2 h_k(x) \approx R_J$  while  $\nabla^2 f(x) \approx J(x)^T J(x)$ .

Another nice feature of  $h_k(x)$  can be seen when considering the case where  $J(x)$  is square, of full rank, and the residual at the solution is zero. This type of problem is often encountered in the solution of simultaneous non-linear equations. Here, we can rescale the residual with a non-singular scaling matrix,  $D$ , as  $\tilde{R} = DR$ . At the minimum,  $R = 0$  so  $\tilde{R} = 0$  as well, and  $\tilde{R} \neq 0$  implies that  $R \neq 0$ . Thus, the two problems are equivalent. Now we have  $\tilde{J} = DJ$

and, since both  $D$  and  $J$  are square matrices of full rank,  $\tilde{J}^+ = J^{-1}D^{-1}$  [99].

This leads to

$$\tilde{J}^+ \tilde{R} = J^{-1}D^{-1}DR = J^+R . \quad (6.8)$$

Hence, the  $h_k(x)$  functions are scale invariant with respect to scale changes in  $R$  for this special case. However, the result for scale changes in  $x$  is different.

Let

$$\tilde{x} = Dx , \quad (6.9)$$

$$\tilde{J} = JD^{-1} , \quad (6.10)$$

$$\tilde{J}^+ = DJ^+ . \quad (6.11)$$

Then,

$$h_k(x) = \frac{1}{2} \|J^+R\|^2 , \quad (6.12)$$

$$= \frac{1}{2} \|D^{-1}DJ^+R\|^2 , \quad (6.13)$$

$$= \frac{1}{2} \|D^{-1}\tilde{J}^+R\|^2 , \quad (6.14)$$

$$\neq \frac{1}{2} \|\tilde{J}^+R\|^2 . \quad (6.15)$$

Thus, the  $h_k(x)$  functions are not scale invariant with respect to scale changes in  $x$ .

There are two problems with these functions that must be solved before global convergence can be proved. First,  $h_k(x)$  does not always decrease

as  $x \rightarrow x^*$ . Some problems have a singular value that goes to zero as  $x \rightarrow x^*$  causing  $h_k(x) \rightarrow \infty$ . The second problem can be seen with the rearrangement

$$h_k(x) = \frac{1}{2}R(x)^T(J(x_k)^+)^T J(x_k)^+ R(x) \quad . \quad (6.16)$$

This makes  $h_k(x)$  look like the square of an alternate norm induced by the matrix

$$A_k = (J(x_k)^+)^T J(x_k)^+ \quad . \quad (6.17)$$

This definition of  $A_k$  is not of full rank; hence, this is not a norm. Herein lies the difficulty: components of  $R(x)$  that are orthogonal to the column space of  $J$  are not considered. An algorithm that minimizes this function could not prevent these components increasing without bound. Hence, these orthogonal directions must also be considered in order to obtain global convergence.

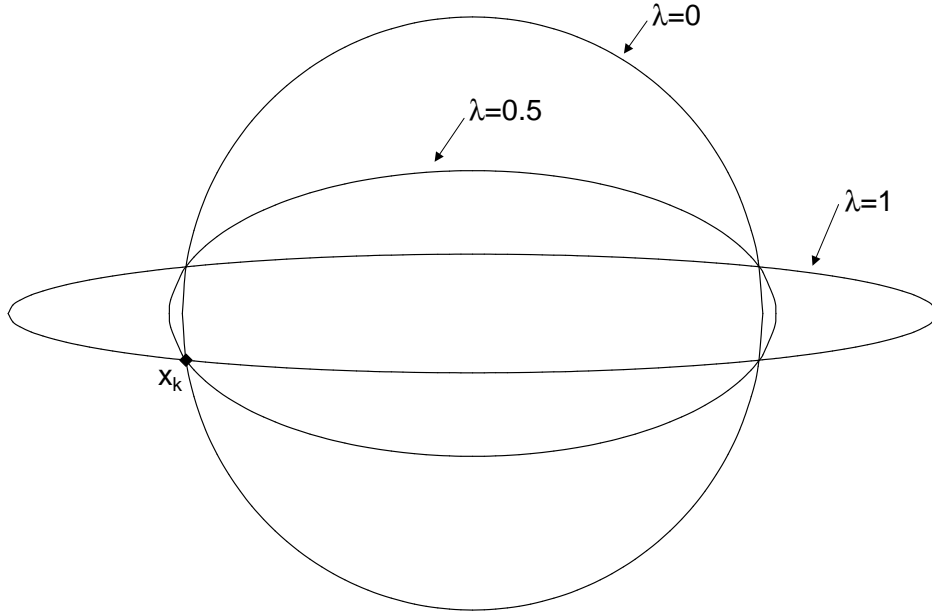
Both of these problems are overcome by defining  $A_k$  as

$$A_k = (1 - \lambda_k)(J_k^+)^T J_k^+ + \lambda_k I, \quad 0 < \lambda_k \leq 1 \quad . \quad (6.18)$$

Then define

$$h_k(x) = \frac{1}{2}R(x)^T A_k R(x) = \frac{1}{2}\|R(x)\|_{A_k}^2 \quad . \quad (6.19)$$

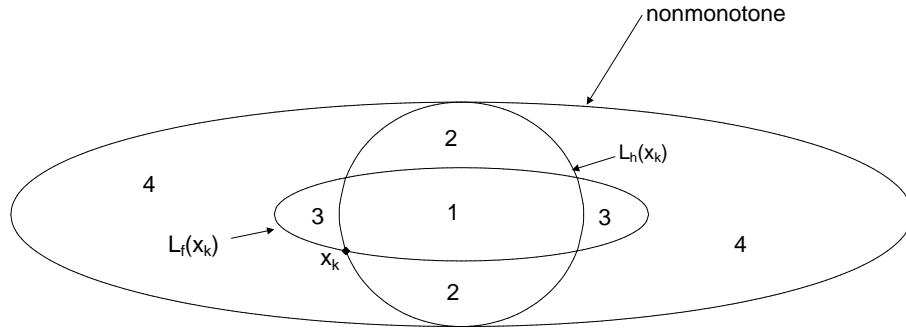
The parameter  $\lambda_k$  interpolates between a minimum distance function obtained at  $\lambda_k = 0$  and the objective function obtained at  $\lambda_k = 1$ . Changing  $\lambda_k$  allows us to adjust the amount of allowable function increase during the line search, Figure 6.1.



**Figure 6.1.** How  $\lambda$  affects the minimum-distance function.

It is also instructive to compare this with the nonmonotone method of Grippo et al. [53]. Suppose  $\max_{k-p \leq i \leq k} (f(x_i))$  is equal to the maximum function value obtainable in  $L_h(x_k)$ . Then, the regions of acceptable next iterates for a quadratic objective function are as shown in Figure 6.2. The minimum distance function will reject points in regions 3 and 4 while the nonmonotone method of Grippo et al. accepts such points. In these regions,  $\|x_k - x^*\|$  is increasing so that the algorithm is diverging. The nonmonotone algorithm

obtains convergence when  $\max_{k-p \leq i \leq k} (f(x_i))$  eventually decreases. The minimum distance function obtains convergence by decreasing either  $\|J_k^+ R(x)\|$  or the objective function.



**Figure 6.2.** Comparison of nonmonotone and minimum distance regions.

In the next section, we consider minimizing a sequence of these  $h_k(x)$  functions using a line search method. However, we first present two lemmas.

**Lemma 6.1** For the Gauss-Newton direction, the above definition for  $h_k(x)$  relaxes the bounded away from orthogonality condition as  $\lambda_k$  decreases from one. That is, the parameter  $\lambda_k$  decreases the minimum allowable angle between the negative gradient and the Gauss-Newton direction as  $\lambda_k$  decreases to zero.

**Proof:** Using the Gauss-Newton direction,  $d_k = -J_k^+ R_k$ ,

$$\cos(\theta) = \frac{R_k^T A_k J_k J_k^+ R_k}{\|J_k^+ R_k\| \|J_k^T A_k R_k\|} . \quad (6.20)$$

Noting that

$$A_k \rightarrow (J_k^+)^T J_k^+ \quad (6.21)$$

as  $\lambda_k$  approaches zero we obtain the limit

$$\cos(\theta) = \frac{R_k^T (J_k^+)^T J_k^+ R_k}{\|J_k^+ R_k\|^2} = 1 , \quad (6.22)$$

while, for  $\lambda_k = 1$  we get the  $\cos(\theta)$  obtained for the Gauss-Newton method. ■

**Lemma 6.2** The above definition for  $h_k(x)$  relaxes the sufficient decrease condition as  $\lambda_k$  decreases from one if a decrease in  $\frac{1}{2} \|R_{k+1}^T (J_k^+)^T J_k^+ R_{k+1}\|^2$  is obtained.

**Proof:** Take the sufficient decrease condition for  $f(x)$  and multiply by  $\lambda_k$  to obtain

$$\frac{\lambda_k}{2} \|R_{k+1}\|^2 \leq \frac{\lambda_k}{2} \|R_k\|^2 + \epsilon \lambda_k R_k^T J_k d_k . \quad (6.23)$$

Also consider

$$\begin{aligned} \frac{1 - \lambda_k}{2} \|R_{k+1}\|_{(J_k^+)^T J_k^+}^2 &\leq \frac{1 - \lambda_k}{2} \|R_k\|_{(J_k^+)^T J_k^+}^2 \\ &+ \epsilon(1 - \lambda_k) \left( J_k^T (J_k^+)^T J_k^+ R_k \right)^T d_k . \end{aligned} \quad (6.24)$$

The sum of equations (6.23) and (6.24) gives the sufficient decrease condition for  $h_k(x)$ . If  $\lambda_k = 1$ , the sufficient decrease condition for  $f(x)$  is obtained. However, if a reduction in  $\frac{1}{2}\|R_{k+1}^T(J_k^+)^T J_k^+ R_{k+1}\|^2$  is obtained, this term can compensate for an increase in  $f(x)$ . Hence, the standard sufficient decrease condition is relaxed if a decrease in  $\frac{1}{2}\|R_{k+1}^T(J_k^+)^T J_k^+ R_{k+1}\|^2$  is obtained. That an increase in  $f(x)$  does happen in some cases is illustrated in the next section using Rosenbrock's function. ■

These lemmas show that the features the minimum distance functions had of relaxing the bounded away and sufficient decrease conditions are retained by this modification.

## 6.1 Line Search Algorithm

We now discuss the algorithm in detail and present the global convergence proof. The algorithm presented here uses the steepest descent search direction. While this is normally not a good choice because of the slow linear rate of convergence often observed, it is quite satisfactory in this case because the steepest descent direction for  $h_k(x)$  is close to the Gauss-Newton direction when  $\lambda_k$  is close to zero. Obviously, the algorithm could be improved by considering other search directions.

**Algorithm 6.3** The minimum distance line-search algorithm is:

Compute  $R(x_1)$ ,  $f(x_1)$ ,  $J(x_1)$ ,  $J^+(x_1)$ ;

Set  $q_1 \geq f(x_1)$ ;

Set  $0 < \epsilon < 1$ .

(1) IF converged THEN quit

(2) Pick  $\lambda_k$  so that

$$\frac{R_k^T (J_k^+)^T J_k^+ R_k}{2(q_k - f_k) + R_k^T (J_k^+)^T J_k^+ R_k} \leq \lambda_k \leq 1 ,$$

$$A_k = (1 - \lambda_k)(J_k^+)^T J_k^+ + \lambda_k I ,$$

$$h_k(x_k) = \frac{1}{2} R_k^T A_k R_k .$$

(3) Obtain the search direction:

$$d_k = - \frac{J_k^T A_k R_k}{\|J_k^T A_k R_k\|} .$$

(4) Pick  $\alpha_k$  to minimize  $h_k(x_k + \alpha_k d_k)$ , then set  $x_{k+1} = x_k + \alpha_k d_k$ . This line-search is designed to ensure that the following sufficient decrease condition is satisfied:

$$h_k(x_{k+1}) \leq h_k(x_k) - \sigma(|\nabla h_k(x_k)^T d_k|) .$$

(5) Pick  $q_{k+1}$  so that:

$$f(x_{k+1}) \leq q_{k+1} \leq q_k + \epsilon [h_k(x_{k+1}) - h_k(x_k)] .$$

(6) Compute  $J(x_{k+1})$  and increment  $k$ .

(7) GOTO step 1

Note that if  $\lambda_k = 1$  at any iteration, our equation for choosing  $q_{k+1}$  leads to the possibility that  $\lambda_k < 1$  for the next iteration. Also, if  $\lambda_k$  is close to zero, but  $R_k^T (J_k^+)^T J_k^+ R_k$  suddenly becomes large, our equation for choosing  $\lambda_k$  yields a value for  $\lambda_k$  closer to one. Thus, the algorithm can dynamically adjust  $\lambda_k$  as needed to switch between the influence of curvature and descent. When  $\lambda_k$  is small, the objective function can increase to allow for the curvature of the problem. When  $\lambda_k$  is close to one, the steepest descent direction for  $f$  is searched emphasizing descent.

We now show that the algorithm is well defined. The sufficient decrease condition for the line search in step 4 can be satisfied by many standard line search algorithms [89]. We also need to show that our choice for  $\lambda_k$  keeps  $f(x_{k+1})$  small enough that a new value for  $q_{k+1}$  can be obtained from the equation

$$f(x_{k+1}) \leq q_{k+1} \leq q_k + \epsilon[h_k(x_{k+1}) - h_k(x_k)] . \quad (6.25)$$

**Theorem 6.4** Suppose  $\lambda_k$  is chosen to satisfy

$$\frac{R_k^T (J_k^+)^T J_k^+ R_k}{2(q_k - f_k) + R_k^T (J_k^+)^T J_k^+ R_k} \leq \lambda_k \leq 1 , \quad (6.26)$$

and the sufficient decrease condition:

$$h_k(x_{k+1}) \leq h_k(x_k) - \sigma(|\nabla h_k(x_k)^T d_k|) \quad (6.27)$$

is satisfied by the line search. Then

$$f(x_{k+1}) \leq q_k + \epsilon[h_k(x_{k+1}) - h_k(x_k)] . \quad (6.28)$$

**Proof:** Starting from the sufficient decrease condition

$$h_k(x_{k+1}) \leq h_k(x_k) - \sigma(|\nabla h_k(x_k)^T d_k|) \quad (6.29)$$

and  $0 < \epsilon < 1$  we have

$$h_k(x_{k+1}) \leq h_k(x_k) + \epsilon[h_k(x_{k+1}) - h_k(x_k)] . \quad (6.30)$$

Using the definition of  $A_k$

$$A_k = (1 - \lambda_k)(J_k^+)^T J_k^+ + \lambda_k I \quad (6.31)$$

and the definition of  $h_k(x_k)$

$$h_k(x_k) = \frac{1}{2} R_k^T A_k R_k , \quad (6.32)$$

we have

$$\frac{1}{2} R_{k+1}^T ((1 - \lambda_k)(J_k^+)^T J_k^+ + \lambda_k I) R_{k+1} \leq h_k(x_k) + \epsilon[h_k(x_{k+1}) - h_k(x_k)] . \quad (6.33)$$

Expanding, and noting the definition for  $f(x_{k+1})$  we obtain

$$\begin{aligned} \lambda_k f(x_{k+1}) &\leq h_k(x_k) - \frac{1 - \lambda_k}{2} R_{k+1}^T (J_k^+)^T J_k^+ R_{k+1} + \\ &\quad \epsilon [h_k(x_{k+1}) - h_k(x_k)] . \end{aligned} \quad (6.34)$$

Again, expanding  $A_k$  in  $h_k(x_k)$ , we get

$$\begin{aligned} \lambda_k f(x_{k+1}) &\leq \frac{1}{2} R_k^T \left( (1 - \lambda_k) (J_k^+)^T J_k^+ + \lambda_k I \right) R_k - \\ &\quad \frac{1 - \lambda_k}{2} R_{k+1}^T (J_k^+)^T J_k^+ R_{k+1} + \\ &\quad \epsilon [h_k(x_{k+1}) - h_k(x_k)] . \end{aligned} \quad (6.35)$$

Rearranging and dividing by  $\lambda_k$  gives

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) + \frac{1 - \lambda_k}{2\lambda_k} R_k^T (J_k^+)^T J_k^+ R_k - \\ &\quad \frac{1 - \lambda_k}{2\lambda_k} R_{k+1}^T (J_k^+)^T J_k^+ R_{k+1} + \\ &\quad \frac{\epsilon}{\lambda_k} [h_k(x_{k+1}) - h_k(x_k)] . \end{aligned} \quad (6.36)$$

Here we are justified in dividing by  $\lambda_k$  since

$$0 < \frac{R_k^T (J_k^+)^T J_k^+ R_k}{2(q_k - f_k) + R_k^T (J_k^+)^T J_k^+ R_k} \leq \lambda_k \quad (6.37)$$

because  $q_k \geq f_k$  was enforced during the previous iteration of the algorithm,

and if  $R_k^T (J_k^+)^T J_k^+ R_k \geq 0$  ever becomes zero, the algorithm terminates. Rear-

ranging the inequality for  $\lambda_k$  gives

$$R_k^T (J_k^+)^T J_k^+ R_k \leq 2\lambda_k (q_k - f_k) + \lambda_k R_k^T (J_k^+)^T J_k^+ R_k , \quad (6.38)$$

or

$$f(x_k) + \frac{1 - \lambda_k}{2\lambda_k} R_k^T (J_k^+)^T J_k^+ R_k \leq q_k . \quad (6.39)$$

Observing that

$$\frac{1 - \lambda_k}{2\lambda_k} R_{k+1}^T (J_k^+)^T J_k^+ R_{k+1} \geq 0 , \quad (6.40)$$

we obtain

$$f(x_k) + \frac{1 - \lambda_k}{2\lambda_k} R_k^T (J_k^+)^T J_k^+ R_k - \frac{1 - \lambda_k}{2\lambda_k} R_{k+1}^T (J_k^+)^T J_k^+ R_{k+1} \leq q_k . \quad (6.41)$$

Inserting this relation into (6.36) we get

$$f(x_{k+1}) \leq q_k + \frac{\epsilon}{\lambda_k} [h_k(x_{k+1}) - h_k(x_k)] . \quad (6.42)$$

Noting that  $0 < \lambda_k \leq 1$  with  $[h_k(x_{k+1}) - h_k(x_k)] < 0$  we obtain the final result

$$f(x_{k+1}) \leq q_k + \epsilon [h_k(x_{k+1}) - h_k(x_k)] \quad (6.43)$$

thus completing the proof. ■

Next, we need to show that minimizing  $h_k(x)$  also minimizes  $f(x)$ .

The following theorem characterizes the stationary point convergence of the algorithm.

**Theorem 6.5** Suppose  $\nabla h_k(x_k) = 0$ . Then  $R(x_k)$  is orthogonal to the column space of  $J(x_k)$ .

**Proof:** We have

$$\nabla h_k(x_k) = J_k^T A_k R_k \quad (6.44)$$

$$= J_k^T \left[ (1 - \lambda_k)(J_k^+)^T J_k^+ + \lambda_k I \right] R_k \quad (6.45)$$

$$= (1 - \lambda_k) J_k^T (J_k^+)^T J_k^+ R_k + \lambda_k J_k^T R_k \quad (6.46)$$

$$= (1 - \lambda_k) \left[ J_k^+ J_k \right]^T J_k^+ R_k + \lambda_k J_k^T R_k \quad (6.47)$$

Using Equation 2.14 (noting that  $R_J = J_k^+ J_k$  is symmetric), we have

$$\nabla h_k(x_k) = (1 - \lambda_k) J(x_k)^+ J(x_k) J(x_k)^+ R(x_k) + \lambda_k J(x_k)^T R(x_k) \quad (6.48)$$

and from Equation 2.11 we obtain the simplified equation

$$\nabla h_k(x_k) = (1 - \lambda_k) J(x_k)^+ R(x_k) + \lambda_k J_k^T R(x_k) \quad (6.49)$$

Using the singular value decomposition for  $J(x_k)$ , we obtain

$$\nabla h_k(x_k) = V \text{diag} \left\{ \frac{1 - \lambda_k}{s_i} + \lambda_k s_i \text{ if } s_i > 0, 0 \text{ otherwise} \right\} U^T R(x_k) \quad (6.50)$$

Since  $0 \leq \lambda_k \leq 1$ , if  $\nabla h_k(x_k) = 0$  then  $R(x_k)$  is orthogonal to the column space of  $J(x_k)$ . ■

The definition for  $A_k$  implies that  $\lambda_k \rightarrow 1$  is a sufficient condition for convergence of the algorithm to a stationary point of  $f(x)$  because  $h_k(x) \rightarrow f(x)$  as  $\lambda_k \rightarrow 1$ . However, Theorem 6.5 shows that this condition is not necessary. We now give our global convergence theorem.

**Theorem 6.6** Suppose at every iteration,  $\lambda_k$  is chosen using the equation

$$\frac{R_k^T (J_k^+)^T J_k^+ R_k}{2(q_k - f_k) + R_k^T (J_k^+)^T J_k^+ R_k} \leq \lambda_k \leq 1, \quad (6.51)$$

the sufficient decrease condition

$$h_k(x_{k+1}) \leq h_k(x_k) - \sigma(|\nabla h_k(x_k)^T d_k|) \quad (6.52)$$

is satisfied by the line-search, and  $q_k$  is chosen using

$$f(x_{k+1}) \leq q_{k+1} \leq q_k + \epsilon[h_k(x_{k+1}) - h_k(x_k)]. \quad (6.53)$$

Then  $\{\nabla f(x_k)\} \rightarrow 0$  and any limit point of  $\{x_k\}$  is a stationary point of  $f(x)$ .

**Proof:** Since  $\epsilon > 0$  and  $\sigma(\cdot)$  is a forcing function, the sufficient decrease condition

$$h_k(x_{k+1}) \leq h_k(x_k) - \sigma(|\nabla h_k(x_k)^T d_k|) \quad (6.54)$$

ensures that  $\epsilon[h_k(x_{k+1}) - h_k(x_k)] < 0$ . Thus,

$$q_{k+1} \leq q_k + \epsilon[h_k(x_{k+1}) - h_k(x_k)]. \quad (6.55)$$

ensures that  $q_{k+1} \leq q_k$  and the sequence  $\{q_k\}$  is a decreasing sequence. Noting that  $f(x_k) \leq q_k$  and that  $f(x) \geq 0$  for least-squares problems we see that the  $q_k$  are bounded below by 0. Thus, the sequence  $\{q_k\}$  is a convergent sequence and the sequence  $\{q_k - q_{k+1}\}$  converges to zero. Rearranging

$$0 \leq f(x_{k+1}) \leq q_{k+1} \leq q_k + \epsilon[h_k(x_{k+1}) - h_k(x_k)] \quad (6.56)$$

we have

$$0 \leq -\epsilon[h_k(x_{k+1}) - h_k(x_k)] \leq q_k - q_{k+1} . \quad (6.57)$$

Rearranging the sufficient decrease condition gives

$$-[h_k(x_{k+1}) - h_k(x_k)] \geq \sigma(|\nabla h_k(x_k)^T d_k|) \quad (6.58)$$

so that

$$0 \leq \epsilon\sigma(|\nabla h_k(x_k)^T d_k|) \leq -\epsilon[h_k(x_{k+1}) - h_k(x_k)] \leq q_k - q_{k+1} . \quad (6.59)$$

Since  $\epsilon > 0$  is a constant, this implies that  $|\nabla h_k(x_k)^T d_k|$  approaches zero in the limit. We are using steepest descent direction for  $h_k$ . Thus, the bounded away from orthogonality condition for our algorithm is

$$|\nabla h_k(x_k)^T d_k| = \|\nabla h_k(x_k)\| \quad (6.60)$$

so that  $\|\nabla h_k(x_k)\|$  goes to zero in the limit. This along with Theorem 6.5 implies that  $\{\nabla f(x_k)\} \rightarrow 0$  so that any limit point of  $\{x_k\}$  is a stationary point of  $f(x)$ . ■

The above theorem ensures that the algorithm converges to a point where the gradient of  $f(x)$  is zero. Stronger results can be obtained using additional assumptions on  $f(x)$ . For example, if the minimum is unique, then

the sequence  $\{x_k\}$  also converges and it converges to the minimum. See also Ortega and Rheinboldt [89] for more details.

However, first order stationary point convergence does not preclude the possibility of the algorithm converging to a local maximum of  $f(x)$ . Grippo et al. [53] proved that Newton's method will not converge to a local maximum. A variation of their proof is given here that applies to more general situations, including Algorithm 6.3.

**Theorem 6.7** Suppose that the sequence  $\{x_k\}$ , generated using

$$x_{k+1} = x_k + \alpha_k d_k \quad , \quad (6.61)$$

is well defined and remains in a set  $S$  where the function  $f(x) : S \rightarrow \Re$  has continuous second derivatives. Also, suppose that  $\{x_k\}$  converges to  $x^* \in S$  where  $x^*$  is a local maximum such that there exists an open sphere  $B(x^*)$  where  $y^T \nabla^2 f(x) y < 0$  for all  $x \in B(x^*)$ ,  $y \neq 0$ . If  $\alpha_k \nabla f(x_k)^T d_k < 0$  for all  $k$ , then  $x^*$  cannot be a local maximum.

**Proof:** Let  $\eta > 0$  be such that  $\lambda_{max}[\nabla^2 f(x)] \leq -\eta$  for all  $x \in B(x^*)$ , where  $\lambda_{max}[H]$  is the maximum eigenvalue of  $H$ . Then, we can write:

$$\begin{aligned} f(x_{k+1}) &= f(x_k) + \nabla f(x_k)^T [x_{k+1} - x_k] + \frac{1}{2} [x_{k+1} - x_k]^T \nabla^2 f(x_k) [x_{k+1} - x_k] + \\ &\quad r([x_{k+1} - x_k]) \\ &= f(x_k) + \alpha_k \nabla f(x_k)^T d_k + \frac{1}{2} [x_{k+1} - x_k]^T \nabla^2 f(x_k) [x_{k+1} - x_k] + \end{aligned}$$

$$\begin{aligned}
& r([x_{k+1} - x_k]) \\
\leq & f(x_k) + \alpha_k \nabla f(x_k)^T d_k - \frac{1}{2} \eta \|x_{k+1} - x_k\|^2 + r([x_{k+1} - x_k])
\end{aligned}$$

for all  $k$ , where  $\lim_{\epsilon \rightarrow 0^+} (r(\epsilon)/\epsilon^2) = 0$ .

It follows from  $\alpha_k \nabla f(x_k)^T d_k \leq 0$ , that for sufficiently large  $k$ ,  $f(x_{k+1}) < f(x_k)$  contradicting the assumption that  $x^*$  is a local maximum.  $\blacksquare$

The strice inequality for the condition  $\alpha_k \nabla f(x_k)^T d_k < 0$  in the above proof ensures that convergence is not obtained in a finite number of iterations. If convergence were obtained in a finite number of iterations, then first order convergence to a local maximum could be obtained. That Algorithm 6.3 does not converge to a local maximum is easily obtained using the above theorem and

$$\nabla f(x_k)^T d_k = - \frac{\nabla f(x_k)^T J_k^T A_k R_k}{\|J_k^T A_k R_k\|} = - \frac{R_k^T J_k J_k^T A_k R_k}{\|J_k^T A_k R_k\|} \quad (6.62)$$

so that  $\alpha_k \nabla f(x_k)^T d_k \leq 0$  because  $\alpha_k > 0$  and  $J_k J_k^T A_k$  is positive semi-definite.

## 7. Examples of Algorithm Performance

Algorithm 6.3 was coded using FORTRAN 77. The singular-value decomposition was used to obtain the required pseudo inverses [48]. While this is a rather computationally intensive method, it is fast on test problems and it adds valuable insight into the performance of the algorithm. Since the minimum-distance functions are sensitive to scale changes in  $x$ , the  $i^{th}$  element of  $x_k$  was scaled using the scale factor

$$S_i = \max_{j=0,k}(\|J_i(x_j)\|) . \quad (7.1)$$

This scaling option is used quite often; see, for example, Moré's Marquardt algorithm [80] and NL2SOL [27, 28]. A method of choosing the two parameters  $\lambda_k$  and  $q_k$  is also needed to implement the algorithm. The smallest possible value of  $\lambda_k$ ,

$$\lambda_k = \frac{R_k^T (J_k^+)^T J_k^+ R_k}{2(q_k - f_k) + R_k^T (J_k^+)^T J_k^+ R_k} , \quad (7.2)$$

and the largest possible value of  $q_k$ ,

$$q_{k+1} = q_k + \epsilon [h_k(x_{k+1}) - h_k(x_k)] , \quad (7.3)$$

with  $\epsilon = 10^{-4}$  were always used in this implementation to encourage the algorithm to use the minimum distance idea as much as possible. The initial value

for  $q_1$  was chosen to yield a starting value of  $\lambda_1 = 0.5$ .

The line search performs backtracking or extrapolation to obtain a three point pattern. The derivative at  $x_k$  is used to perform backtracking. Once a three point pattern is obtained, quadratic interpolation is used to obtain an improved point. The line search is terminated at the first point that satisfies

$$h_k(x_{k+1}) \leq h_k(x_k) + \epsilon \alpha_k \nabla h_k(x_k)^T d_k \quad (7.4)$$

where  $\alpha_k$  is the step length.

The termination criteria used in the algorithm are:

- (1)  $f(x_k) \leq 10^{-13}$ , INFO = 1.
- (2)  $\|\nabla f(x_k)\| \leq 10^{-12}$ , INFO = 2.
- (3) The step length  $\leq \max(1, \|x_k\|)10^{-7}$ , INFO = 3.
- (4) The number of function evaluations exceeds  $200(m + 1)$ , INFO = 4  
[82].
- (5)  $\lambda_k \geq 0.9999$ , INFO = 9999.

The last termination test is used to stop the algorithm if it is essentially steepest descent. This was done to prevent the algorithm from consuming too much computer time. When  $\lambda_k$  is too close to one, the minimum distance idea is not working and neither is the steepest descent method.

The algorithm was tested using the Moré, Garbow, Hillstrom test

problems [84, 83]. The performance of the algorithm on these standard problems is given in Tables 7.1 and 7.2. The acronyms used by Moré et al. are given in parentheses for reference. Any missing entries in one of the tables indicate that the algorithm ran into some numerical difficulty and could not continue. A listing of the objective functions is given in Appendix A. The scale factor listed is used to scale the starting point. Moré et al. designed these test problems specifically to test, among other things, how well an algorithm performs under scale changes in  $x$ . In the test set, many problems are repeated with the starting point scaled by 10 and then 100. One example is Rosenbrock's function, problem number 4, where the first row in the table uses the standard initial point  $(-1.2, 1)$ . The two succeeding rows use  $(-12, 10)$  and  $(-120, 100)$  respectively.

For comparison, Tables 7.3 and 7.4 give the performance of an old (mid 1980's) version of Moré's Marquardt algorithm on the same test set [80].

More's convergence criteria are:

- (1) Both actual and predicted relative reductions in the sum-of-squares are at most  $10^{-8}$ , INFO = 1.
- (2) Relative error between two consecutive iterates is at most  $10^{-8}$ , INFO = 2.
- (3) Conditions for both 1 and 2 hold, INFO = 3.

- (4) The cosine of the angle between  $R(x)$  and any column of the Jacobian is zero, INFO = 4.
- (5) Maximum number of function evaluations has been exceeded, INFO = 5.

Problem Number (NPROB)	M	N	Scale Factor	Function Calls (NFEV)	Jacobian Calls (NJEF)	INFO	Objective Function Norm (FINAL L2 NORM)
1	5	10	1	2	2	2	2.2360680D+00
1	5	50	1	2	2	2	6.7082039D+00
2	5	10	1	2	2	2	1.4638501D+00
2	5	50	1	3	3	2	3.4826302D+00
3	5	10	1	2	2	2	1.9097274D+00
3	5	50	1	3	3	2	3.6917294D+00
4	2	2	1	6	6	1	4.3299800D-08
4	2	2	10	5	5	1	6.9715345D-10
4	2	2	100	5	5	1	6.5081163D-10
5	3	3	1	10	9	1	1.9467384D-12
5	3	3	10	26	17	1	1.8631818D-13
5	3	3	100	181	64	1	5.4240424D-10
6	4	4	1	13	13	1	2.7555867D-07
6	4	4	10	16	16	1	3.3882791D-07
6	4	4	100	20	20	1	1.3249207D-07
7	2	2	1	16	9	9999	6.9988753D+00
7	2	2	10	22	12	9999	7.8965477D+00
7	2	2	100				
8	3	15	1	7	7	2	9.0635960D-02
8	3	15	10	29	13	3	9.0635960D-02
8	3	15	100	3	3	9999	4.1747279D+00
9	4	11	1	42	37	3	1.7535838D-02
9	4	11	10	8	7	9999	1.1500400D-01
9	4	11	100	5	3	9999	2.2209595D-01
10	3	16	1	16	12	3	9.3779451D+00
10	3	16	10	16	12	3	9.3779451D+00

**Table 7.1.** Minimum distance algorithm performance on standard test problems.

Problem Number (NPROB)	M	N	Scale Factor	Function Calls (NFEV)	Jacobian Calls (NJEF)	INFO	Objective Function Norm (FINAL L2 NORM)
11	6	31	1	86	45	3	4.7829594D-02
11	6	31	10	17	17	2	4.7829594D-02
11	6	31	100	21	20	2	4.7829594D-02
11	9	31	1	426	181	2	1.1831146D-03
11	9	31	10	22	15	2	1.1831146D-03
11	9	31	100	19	17	2	1.1831146D-03
11	12	31	1	2600	1116	4	2.3497279D-02
11	12	31	10	13	11	2	2.1731040D-05
11	12	31	100	69	33	2	2.1731040D-05
12	3	10	1	219	80	1	2.7028459D-12
13	2	10	1	9	5	9999	1.2257774D+01
14	4	20	1	1001	395	4	3.2677102D+02
14	4	20	10	1001	416	4	3.3237608D+02
14	4	20	100	1000	411	4	3.2367148D+02
15	1	8	1	1	1	2	1.8862380D+00
15	1	8	10	28	28	2	1.8842482D+00
15	1	8	100	36	36	2	5.8315069D+02
15	8	8	1	73	25	9999	8.4654539D-02
15	9	9	1	17	11	1	1.9295463D-09
15	10	10	1	137	39	9999	1.4342183D-01
16	10	10	10	17	4	9999	3.6390336D-01
16	10	10	100	11	10	2	6.1315453D-07
16	30	30	1	11	7	1	3.5793833D-08
16	40	40	1	11	7	1	3.7277429D-08
17	5	33	1	17	13	2	7.3924926D-03
18	11	65	1	25	19	3	2.0034404D-01

**Table 7.2.** Minimum distance algorithm performance on standard test problems continued.

Problem Number (NPROB)	M	N	Scale Factor	Function Calls (NFEV)	Jacobian Calls (NJEF)	INFO	Objective Function Norm (FINAL L2 NORM)
1	5	10	1	3	2	3	2.2360680D+00
1	5	50	1	3	2	2	6.7082039D+00
2	5	10	1	3	2	1	1.4638501D+00
2	5	50	1	3	2	1	3.4826302D+00
3	5	10	1	3	2	1	1.9097274D+00
3	5	50	1	3	2	1	3.6917294D+00
4	2	2	1	21	16	4	0.0000000D+00
4	2	2	10	8	5	2	0.0000000D+00
4	2	2	100	6	4	2	0.0000000D+00
5	3	3	1	11	8	2	9.9365231D-17
5	3	3	10	20	15	2	1.0446809D-19
5	3	3	100	19	16	2	3.7665334D-29
6	4	4	1	500	499	5	0.0000000D+00
6	4	4	10	500	499	5	0.0000000D+00
6	4	4	100	66	65	4	9.3220945D-35
7	2	2	1	14	8	1	6.9988752D+00
7	2	2	10	19	12	1	6.9988752D+00
7	2	2	100	24	17	1	6.9988752D+00
8	3	15	1	6	5	1	9.0635960D-02
8	3	15	10	37	36	1	4.1747687D+00
8	3	15	100	14	13	1	4.1747687D+00
9	4	11	1	18	16	1	1.7535838D-02
9	4	11	10	78	70	1	3.2052193D-02
9	4	11	100	500	369	5	2.2569255D-02
10	3	16	1	126	116	2	9.3779451D+00
10	3	16	10	400	346	5	7.9607763D+02

**Table 7.3.** Moré’s Marquardt algorithm performance on standard test problems.

Problem Number (NPROB)	M	N	Scale Factor	Function Calls (NFEV)	Jacobian Calls (NJEF)	INFO	Objective Function Norm (FINAL L2 NORM)
11	6	31	1	8	7	1	4.7829594D-02
11	6	31	10	14	13	1	4.7829594D-02
11	6	31	100	15	14	1	4.7829594D-02
11	9	31	1	8	7	2	1.1831146D-03
11	9	31	10	19	15	1	1.1831146D-03
11	9	31	100	18	15	1	1.1831146D-03
11	12	31	1	10	9	3	2.1731040D-05
11	12	31	10	13	12	2	2.1731040D-05
11	12	31	100	34	28	3	2.1731040D-05
12	3	10	1	7	6	2	1.5700924D-16
13	2	10	1	21	12	1	1.1151779D+01
14	4	20	1	254	236	1	2.9295429D+02
14	4	20	10	53	42	1	2.9295429D+02
14	4	20	100	238	222	1	2.9295429D+02
15	1	8	1	1	1	4	1.8862380D+00
15	1	8	10	29	28	1	1.8842482D+00
15	1	8	100	47	46	1	1.8842482D+00
15	8	8	1	39	20	1	5.9303235D-02
15	9	9	1	12	9	2	1.9859084D-16
15	10	10	1	25	12	1	8.0647100D-02
16	10	10	1	14	12	2	3.0967064D-15
16	10	10	10	13	8	2	3.0303191D-15
16	10	10	100	22	20	2	2.1868857D-15
16	30	30	1	19	14	2	2.2480051D-13
16	40	40	1	19	14	2	4.7146716D-14
17	5	33	1	18	15	1	7.3924926D-03
18	11	65	1	16	12	1	2.0034404D-01

**Table 7.4.** Moré’s Marquardt algorithm performance on standard test problems, continued.

The algorithm performs well on problems where  $\|J(x_k)^+R(x_k)\| \rightarrow 0$  in some neighborhood of the minimum. When this condition does not exist,  $\lambda_k \rightarrow 1$  and the algorithm becomes steepest descent. Hence, the algorithm performs poorly on such problems.

While these results are not spectacular they are encouraging. We did not expect to get exceptional results using the steepest descent direction. Incorporating a better search direction into the algorithm should help to greatly improve algorithm performance. The results indicate that more work could be done to improve the scaling of our algorithm. However, many of the desired objectives of the above analysis were observable in the test cases. Specific examples taken from some of the test problems are listed in the following sections.

## 7.1 Rosenbrock's Function

The residuals for Rosenbrock's function [102], problem number 4, are

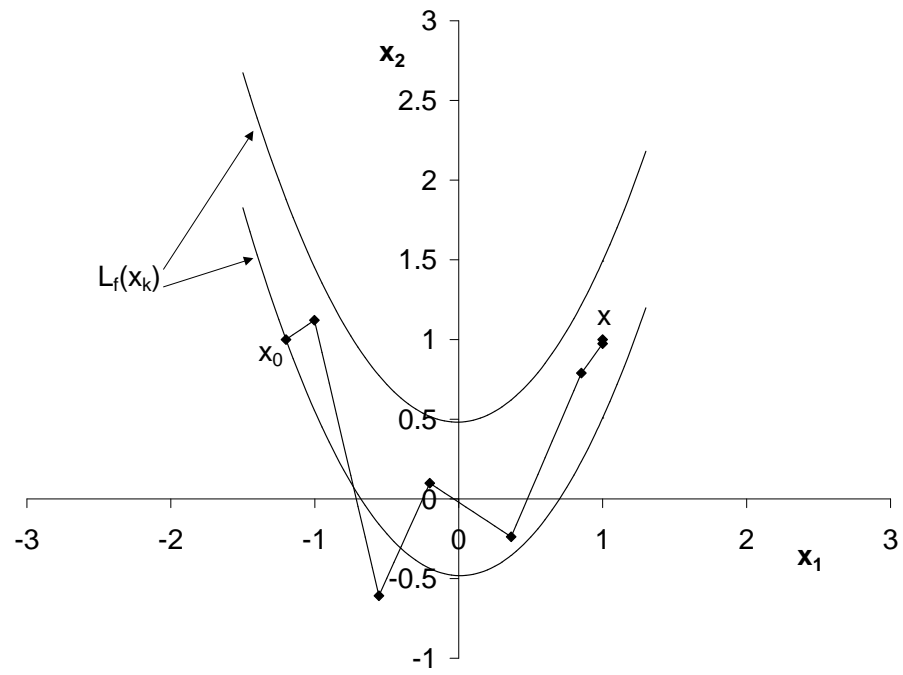
$$r_1(x) = 10(x_2 - x_1^2) \tag{7.5}$$

$$r_2(x) = 1 - x_1 \tag{7.6}$$

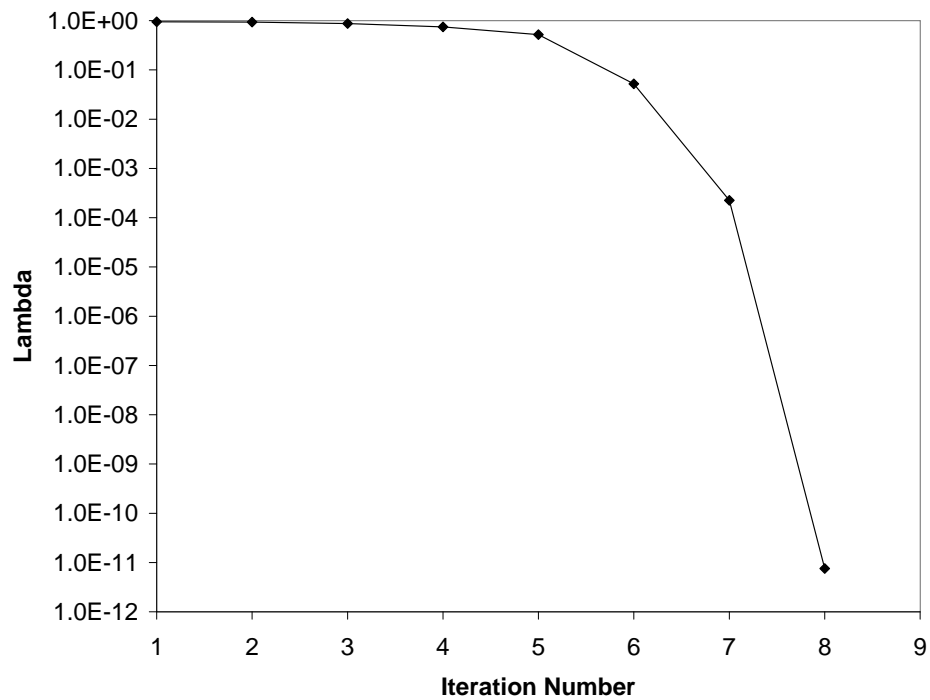
Rosenbrock's function is a classic zero-residual least-squares problem where  $\|J(x_k)^+R(x_k)\| \rightarrow 0$  and Gauss-Newton exhibits a quadratic convergence rate.

This problem illustrates several nice properties of the minimum-distance

algorithm. First, if the algorithm is initialized using a value of  $\lambda_1$  near one, then  $\lambda_k \rightarrow 0$  as  $x_k \rightarrow x^*$ . To illustrate this, we ran the algorithm with  $\lambda_1 = 0.95$ . The path taken by the algorithm in this case is given in Figure 7.1. As can be seen, the first direction is essentially the negative-gradient direction. However, succeeding directions quickly move away from the negative-gradient direction and the algorithm quickly converges to the minimum. Also observable in the early iterations of Figure 7.1 are the classic oscillations characteristic of using a steepest descent algorithm. These oscillations dampen as  $\lambda_k$  gets smaller and the Gauss-Newton direction is approached. That  $\lambda_k$  quickly becomes small is evident from Figure 7.2, a semi-log plot of  $\lambda_k$  versus the iteration number.

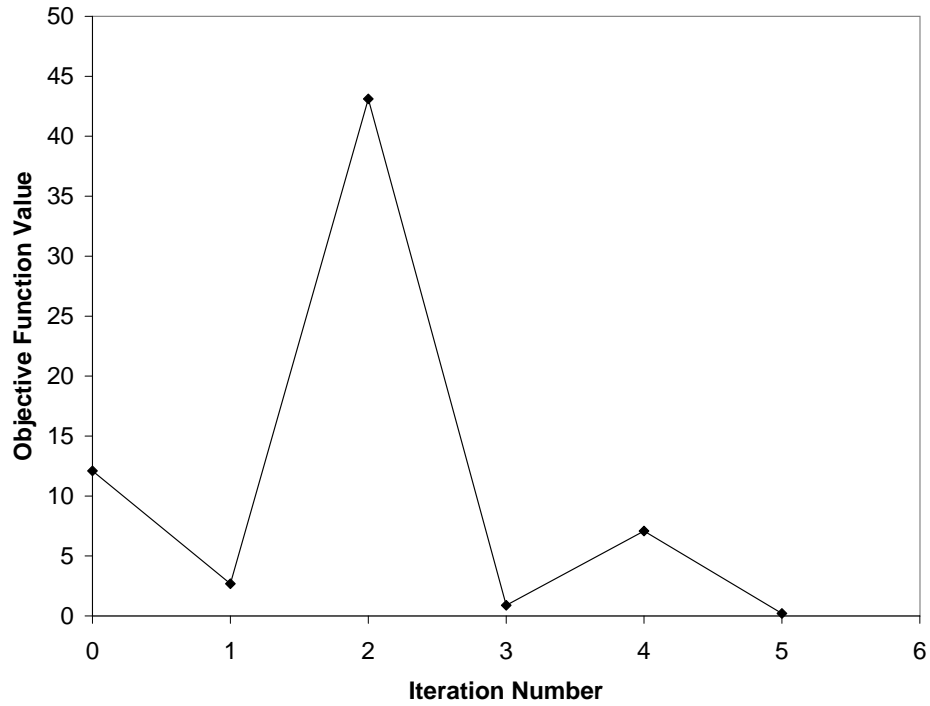


**Figure 7.1.** Minimum-distance function path for  $\lambda_1 = 0.95$ .



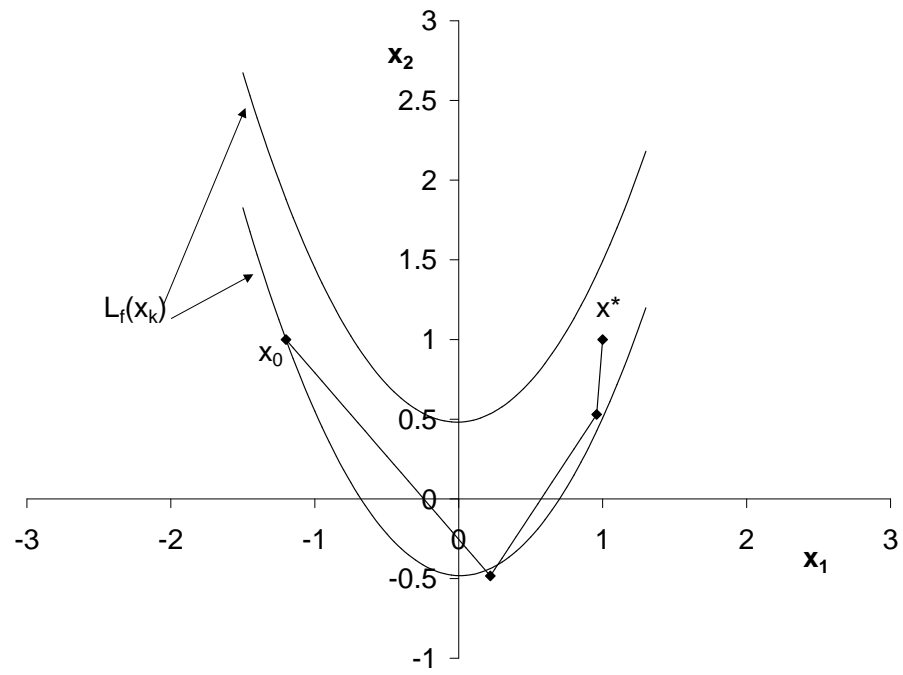
**Figure 7.2.** Convergence of  $\lambda_k$  for  $\lambda_1 = 0.95$ .

This problem also illustrates that values of  $\lambda_k$  that are not too close to one allow the objective function to increase, yet convergence to the minimum is still obtained. For this  $\lambda_1 = 0.95$  case, this occurs at iteration two as shown in Figure 7.3.



**Figure 7.3.** Objective function values for  $\lambda_1 = 0.95$ .

Next, the algorithm was run with  $\lambda_1 = 0.8$ . This yields a starting direction that is closer to the Gauss-Newton direction as seen in Figure 7.4. This allows  $\lambda_k$  to decrease more quickly, Figure 7.5. This quicker drop in  $\lambda_k$  allows the algorithm to increase the objective function, Figure 7.6. These combined effects help speed up the algorithm resulting in fewer iterations being required than in the  $\lambda_1 = 0.95$  case.



**Figure 7.4.** Minimum-distance function path for  $\lambda_1 = 0.8$ .

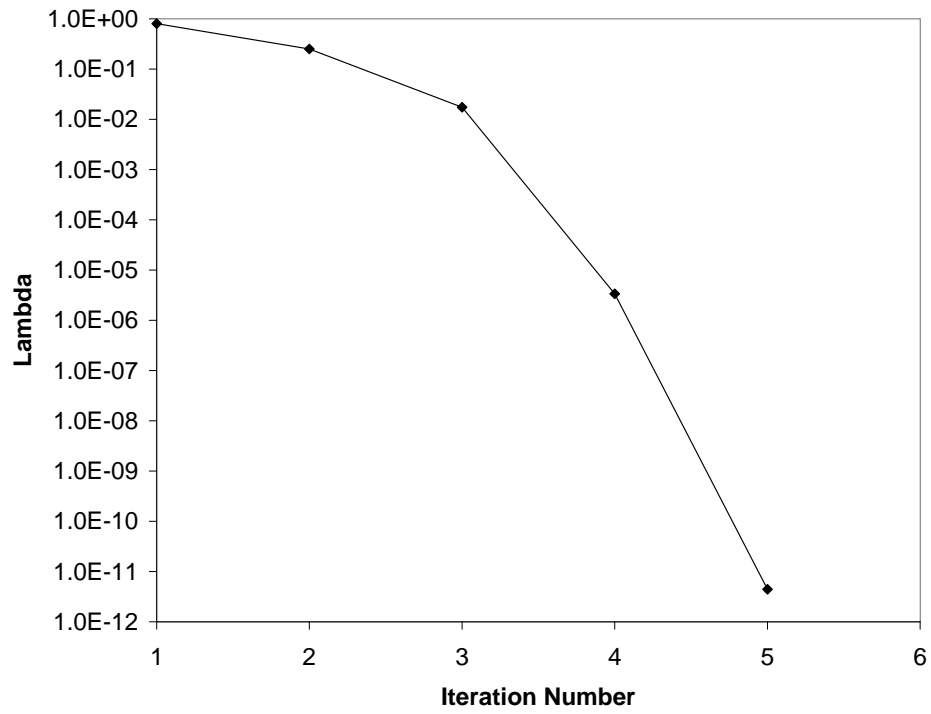
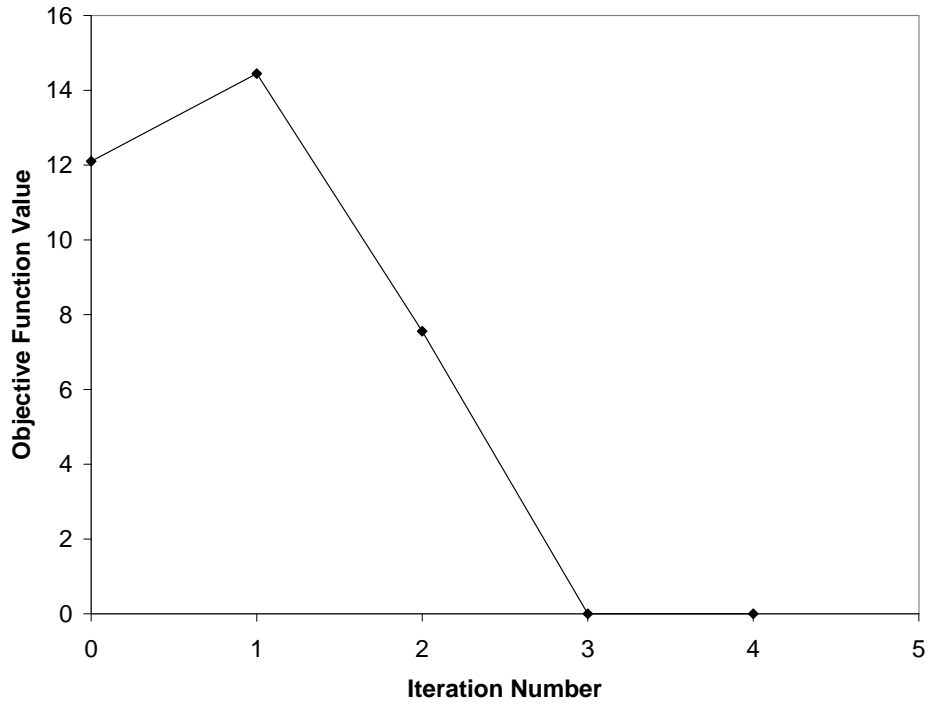


Figure 7.5. Convergence of  $\lambda_k$  for  $\lambda_1 = 0.8$ .



**Figure 7.6.** Objective function values for  $\lambda_1 = 0.8$ .

As  $\lambda_1$  gets smaller, greater increases in  $f(x)$  are allowed. This is illustrated with the same series of plots where the algorithm is run with  $\lambda_1 = 0.5$ . Here, a starting direction very near the Gauss-Newton direction is obtained, Figure 7.7, and  $\lambda_k$  very quickly gets small, Figure 7.8. As in the previous examples, the objective function is allowed to increase yet eventually converges to zero, Figure 7.9. This final case obtains the minimum in fewer iterations than required for the previous two cases.

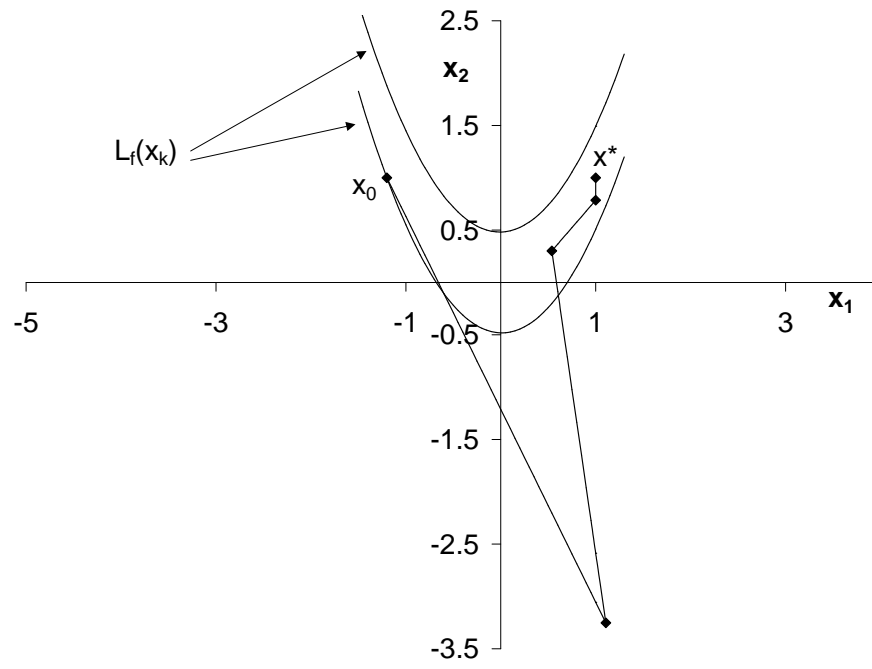


Figure 7.7. Minimum-distance function path for  $\lambda_1 = 0.5$ .

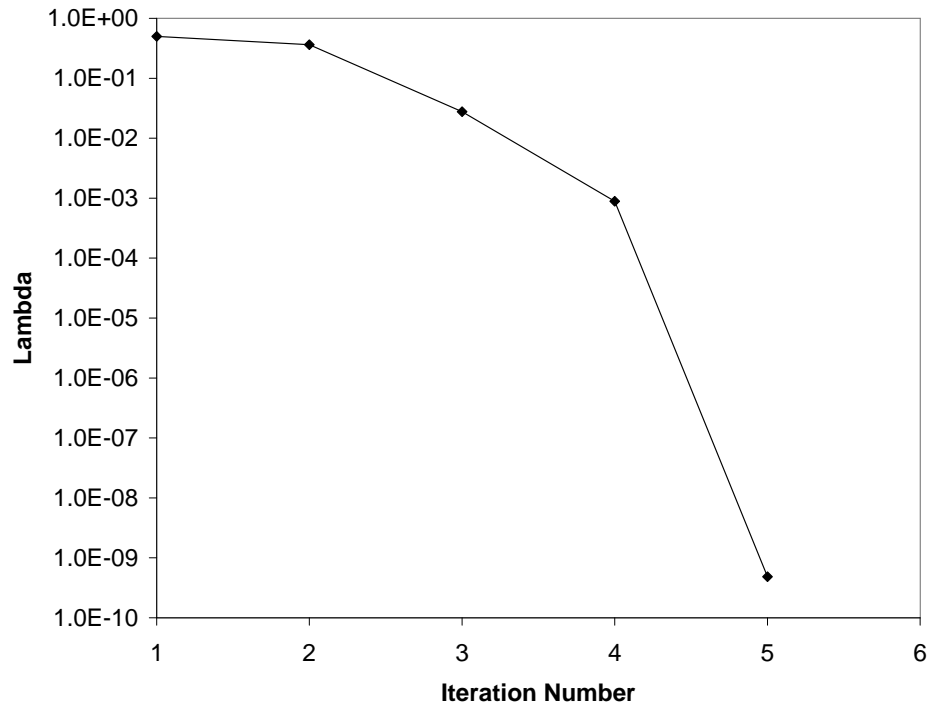


Figure 7.8. Convergence of  $\lambda_k$  for  $\lambda_1 = 0.5$ .

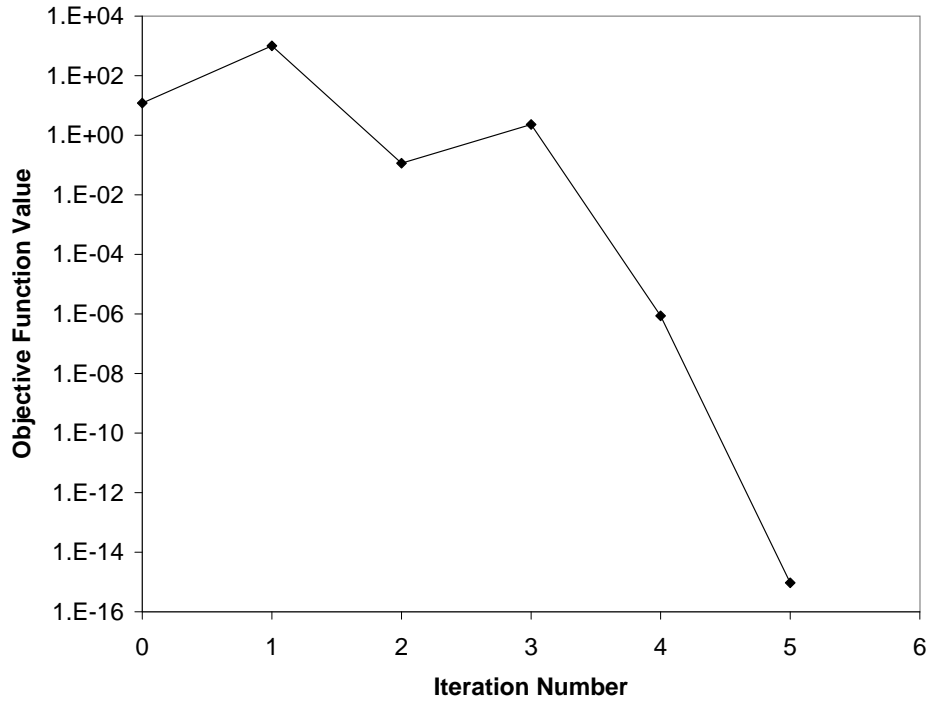


Figure 7.9. Objective function values for  $\lambda_1 = 0.5$ .

## 7.2 Freudenstein and Roth's Function

The residuals for the Freudenstein and Roth function [44], problem number 7, are

$$r_1(x) = -13 + x_1 + ((5 - x_2)x_2 - 2)x_2 \quad (7.7)$$

$$r_2(x) = -29 + x_1 + ((x_2 + 1)x_2 - 14)x_2 \quad (7.8)$$

While the minimum-distance function algorithm does very well on Rosenbrock's function, the Freudenstein and Roth function causes it quite some difficulty. This problem has a non-zero residual for one of its local minimums where one of the singular values approaches zero as  $\{x_k\} \rightarrow x^*$ . The Jacobian for this problem is

$$J(x) = \begin{bmatrix} 1 & -3x_2^2 + 10x_2 - 2 \\ 1 & 3x_2^2 + 2x_2 - 14 \end{bmatrix} . \quad (7.9)$$

This Jacobian becomes singular when the first column is a scalar multiple of the second column. That is, the Jacobian becomes singular when

$$-3x_2^2 + 10x_2 - 2 = 3x_2^2 + 2x_2 - 14 . \quad (7.10)$$

This is a quadratic equation in  $x_2$  that can be easily solved to obtain  $x_2 = -0.8968\dots$  and  $x_2 = 2.230138\dots$ . The local minimum that causes the difficulty is at  $(11.41\dots, -0.8968\dots)$ . The minimum-distance algorithm has trouble when  $\|J(x_k)^+R(x_k)\| \rightarrow \infty$ . In this case, this happens when  $x_2 \rightarrow -0.8968\dots$ . Once this happens,  $\lambda_k \rightarrow 1$ , the steepest-descent algorithm is obtained and the algorithm terminates when  $\lambda_k \geq 0.9999$ .

As seen in Table 7.1, the first instance of problem number 7 terminated at a point very near the local minimum where the norm of the objective function obtained is very near the optimal value of  $6.99887\dots$ . For this case, the minimum-distance algorithm took only a few iterations to get this close.

Once this point is reached, the resulting steepest-descent algorithm is unable to successfully find the minimum in a reasonable number of iterations.

The results for the other two cases, where the algorithm is started at 10 and 100 times farther away from the minimum, were not as good. During the first few iterations large decreases in the objective function and also in  $\|J(x_k)^+R(x_k)\|$  were obtained causing  $\lambda_k$  to get very small. Then, the algorithm starts to get close to  $x_2 = -0.8968\dots$  where the associated singular value is close to zero. This causes  $\|J(x_k)^+R(x_k)\|$  to increase as expected. However, at this point, an interesting scaling problem is observed. The drop in the singular value is enough to dominate the Gauss-Newton direction but not enough to affect  $\lambda_k$ . The drop obtained in the objective function during the first few iterations caused the  $2(q_k - f_k)$  term in the equation for obtaining  $\lambda_k$ ,

$$\lambda_k = \frac{R_k^T (J_k^+)^T J_k^+ R_k}{2(q_k - f_k) + R_k^T (J_k^+)^T J_k^+ R_k} , \quad (7.11)$$

to become very large. Once the singular value started to become small, this equation should have caused  $\lambda_k$  to approach one causing the algorithm to turn toward the minimum. This is what happened in the first case. In the other cases, the  $2(q_k - f_k)$  term dominates causing  $\lambda_k$  to remain small. This causes the algorithm to search primarily along  $x_2$  making very little progress along  $x_1$ . Eventually, the singular value gets small enough that  $\lambda_k$  approaches one.

However, no progress is made toward the minimum along the other singular direction characterized by  $x_2 = -0.8968\dots$  and steepest-descent cannot find the minimum.

Thus, it is observed that the  $2(q_k - f_k)$  term must be scaled properly for the algorithm to work well. If  $2(q_k - f_k)$  is too large, then  $\lambda_k$  will remain small. If  $2(q_k - f_k)$  is too small, then  $\lambda_k$  will stay close to one. Picking  $\lambda_1$  to be some reasonable value, e.g. 0.5, appears to scale the  $2(q_k - f_k)$  term for many problems. However, some dynamic scaling is needed to improve the minimum-distance algorithm's performance. Such a scaling algorithm needs to find an effective balance between allowing  $\lambda_k \rightarrow 0$  for fast convergence and  $\lambda_k \rightarrow 1$  when a singular value is approaching zero.

### 7.3 Powell's Singular Function

The residuals for Powell's singular function [92], problem number 6, are

$$r_1(x) = x_1 + 10x_2 \tag{7.12}$$

$$r_2(x) = \sqrt{5}(x_3 - x_4) \tag{7.13}$$

$$r_3(x) = (x_2 - 2x_3)^2 \tag{7.14}$$

$$r_4(x) = \sqrt{10}(x_1 - x_4) \tag{7.15}$$

with a minimum objective function value of zero at  $(0, 0, 0, 0)$ . The Jacobian for this problem,

$$J(x) = \begin{bmatrix} 1 & 10 & 0 & 0 \\ 0 & 0 & \sqrt{5} & -\sqrt{5} \\ 2(x_2 - 2x_3) & -4(x_2 - 2x_3) & 0 & 0 \\ \sqrt{10} & 0 & 0 & -\sqrt{10} \end{bmatrix}, \quad (7.16)$$

is obviously singular at the minimum by simply examining the third row. Hence, this problem is a mixture of the above two problems. It is a zero residual problem like Rosenbrock's function but it also has a singular value that approaches zero as  $x_k$  approaches the minimum. The minimum distance function performs very well on this problem since  $\|J(x_k)^+ R(x_k)\| \rightarrow 0$  as  $x_k$  approaches the minimum. This allows  $\lambda_k$  to approach zero so that the algorithm searches a direction approaching the Gauss-Newton direction.

## 8. Extensions and Future Research

Two convergence conditions, the bounded away from orthogonality and sufficient decrease conditions, were identified as keys to algorithm performance. At first, these conditions appeared to be general enough to cover most any efficient algorithm. However, a detailed study of the nonmonotone algorithms in the literature indicated that relaxing these two convergence conditions could yield faster convergence. Further analysis revealed that an estimate of the distance to the minimum relaxes both the bounded away and sufficient decrease conditions. These minimum distance functions also pick an optimal point for a line-search in the following sense. If the estimate of the distance to the minimum is accurate, the distance to the minimum is minimized at each iteration.

We developed a line-search algorithm, based on minimizing a minimum-distance function along its steepest descent direction. Unlike the nonmonotone algorithms in the literature, this minimum distance function algorithm dynamically adjusts to account for changes between the influence of curvature and descent. The algorithm does not force a reduction in the objective function

at each iteration, yet it allows  $f(x)$  to increase during an iteration only if this will result in faster convergence. A variant of this algorithm was tested using some standard test functions. While good performance of the algorithm was observed, faster convergence could be obtained if a better search direction is used. One such enhancement would be to include trust region logic. Trust region algorithms have proven to be very effective in practice and were shown in this thesis to relax the bounded away from orthogonality condition. Thus, if  $\lambda_k \rightarrow 1$ , then an efficient algorithm that relaxes the bounded away from orthogonality condition will result.

The nonmonotone idea of Grippo et. al. [53] has been used for constrained optimization to help eliminate the Maratos effect [8]. This suggests that extending the minimum distance function idea to constrained optimization problems could be beneficial. First, let us consider linear equality constraints,  $e(x) = 0$ . Let  $J^{(+)}$  represent the pseudo inverse of  $J$  restricted to the linear equality constraints,  $e(x) = 0$  [99]. Then the nonlinear least-squares minimum-distance function becomes

$$h_k(x) = \frac{1}{2} \|J^{(+)}(x_k)^+ R(x)\|^2 . \quad (8.1)$$

This is equivalent to the problem of minimizing

$$h_k(x) = \frac{1}{2} \|x - x_k^*\|^2 \quad (8.2)$$

where

$$x_k^* = \operatorname{argmin} \frac{1}{2} \|J(x_k)[x - x_k] + R(x_k)\|^2 \text{ S.T. } e(x) = 0 \quad . \quad (8.3)$$

In our minimum-distance function approach, we minimized an estimate of the distance to the minimum of an unconstrained problem. Here, we use a quadratic approximation to obtain an estimate of the distance to a constrained minimum and we minimize this distance. Such a minimum-distance function would allow feasible and non-feasible iterates.

This idea is very similar to the original idea of allowing  $f(x)$  to increase along the line-search direction in order to make progress along an eigenvector direction with small associated eigenvalue. Here, the algorithm is allowed to go infeasible in order to make progress along a constraint. Consider modifying our least-squares objective function to obtain the penalty function

$$f(x) = Me(x)^2 + \frac{1}{2} \sum_{i=1}^n r_i(x)^2 \quad , \quad (8.4)$$

where  $M$  is a large positive constant. This yields an illconditioned unconstrained nonlinear least-squares problem with an eigenvector (with a small associated singular value) oriented along the constraint  $e(x)$ . If the minimum-distance function idea is applied to this penalty function, the algorithm will allow  $f(x)$  to increase in order to move along the small singular-value eigenvector direction that points along the constraint. Thus, the idea of increasing

$f(x)$  to make progress along a small singular-value direction and the idea of allowing the iterates to go infeasible in order to make progress along a constraint, in this case, coincide. Comparing the performance of the minimum distance idea using the penalty function approach to the performance of the above constrained minimum-distance function could be quite enlightening.

The above example used a penalty function to convert a constrained problem to an unconstrained problem. The minimum-distance function idea also converts a constrained problem into an unconstrained problem. At each function evaluation of this new  $h_k(x)$  we solve a constrained-quadratic subproblem. This idea can easily be extended to include general nonlinear constraints by minimizing

$$h_k(x) = \frac{1}{2} \|x - x_k^*\|^2 \quad , \quad (8.5)$$

where

$$x_k^* = \operatorname{argmin} \frac{1}{2} \|J(x_k)[x - x_k] + R(x_k)\|^2 \text{ S.T. the constraints } \quad . \quad (8.6)$$

If these constraints are linearized, a quadratic problem with linear constraints is obtained that can be solved iteratively to yield the solution to the subproblem. This decouples the nonlinearity of the objective function from the nonlinearity of the constraints. The resulting algorithm should exhibit a fast convergence

rate and the nonmonotone idea incorporated into the minimum-distance function should eliminate the Maratos effect.

## APPENDIX A Standard Test Functions

- (1) Linear function, full rank [84].

$m$  is variable,  $n \geq m$

$$r_i(x) = x_i - \frac{2}{n} \left( \sum_{j=1}^m x_j \right) - 1, 1 \leq i \leq m$$

$$r_i(x) = -\frac{2}{n} \left( \sum_{j=1}^m x_j \right) - 1, m < i \leq n$$

$$x_0 = (1, \dots, 1)$$

$$f = n - m \text{ at } (-1, \dots, -1)$$

- (2) Linear function, rank 1 [84].

$m$  is variable,  $n \geq m$

$$r_i(x) = i \left( \sum_{j=1}^m j x_j \right) - 1$$

$$x_0 = (1, \dots, 1)$$

$$f = \frac{n(n-1)}{2(2n+1)} \text{ at any point where } \sum_{j=1}^m j x_j = \frac{3}{2n+1}$$

- (3) Linear function, rank 1 with zero columns and rows [84].

$m$  is variable,  $n \geq m$

$$r_1(x) = -1$$

$$r_i(x) = (i-1) \left( \sum_{j=2}^{m-1} j x_j \right) - 1, 2 \leq i < n$$

$$r_n(x) = -1$$

$$x_0 = (1, \dots, 1)$$

$$f = \frac{n^2+2n-6}{2(2n-3)} \text{ at any point where } \sum_{j=2}^{m-1} j x_j = \frac{3}{2n-3}$$

- (4) Rosenbrock's Function [102].

$$m = 2, n = 2$$

$$r_1(x) = 10(x_2 - x_1^2)$$

$$r_2(x) = 1 - x_1$$

$$x_0 = (-1.2, 1)$$

$$f = 0 \text{ at } (1, 1)$$

(5) Helical valley function [43].

$$m = 3, n = 3$$

$$r_1(x) = 10[x_3 - 10\theta(x_1, x_2)]$$

$$r_2(x) = 10[\sqrt{x_1^2 + x_2^2} - 1]$$

$$r_3(x) = x_3$$

where

$$\theta(x_1, x_2) = \begin{cases} \frac{1}{2\pi} \arctan\left(\frac{x_2}{x_1}\right), & \text{if } x_1 > 0 \\ \frac{1}{2\pi} \arctan\left(\frac{x_2}{x_1}\right) + 0.5, & \text{if } x_1 < 0 \end{cases}$$

$$x_0 = (-1, 0, 0)$$

$$f = 0 \text{ at } (1, 0, 0)$$

(6) Powell singular function [92].

$$m = 4, n = 4$$

$$r_1(x) = x_1 + 10x_2$$

$$r_2(x) = \sqrt{5}(x_3 - x_4)$$

$$r_3(x) = (x_2 - 2x_3)^2$$

$$r_4(x) = \sqrt{10}(x_1 - x_4)$$

$$x_0 = (3, -1, 0, 1)$$

$$f = 0 \text{ at } (0, 0, 0, 0)$$

(7) Freudenstein and Roth Function [44].

$$m = 2, n = 2$$

$$r_1(x) = -13 + x_1 + ((5 - x_2)x_2 - 2)x_2$$

$$r_2(x) = -29 + x_1 + ((x_2 + 1)x_2 - 14)x_2$$

$$x_0 = (0.5, -2)$$

$$f = 0 \text{ at } (5, 4)$$

$$f = 48.9842 \dots \text{ at } (11.41 \dots, 0.8968 \dots)$$

(8) Bard Function [2].

$$m = 3, n = 15$$

$$r_i(x) = y_i - \left( x_1 + \frac{i}{(16-i)x_2 + \min(i, 16-i)x_3} \right)$$

where

i	$y_i$	i	$y_i$	i	$y_i$
1	0.14	6	0.32	11	0.73
2	0.18	7	0.35	12	0.96
3	0.22	8	0.39	13	1.34
4	0.25	9	0.37	14	2.10
5	0.29	10	0.58	15	4.39

$$x_0 = (1, 1, 1)$$

$$f = 8.21487 \dots 10^{-3}$$

$$f = 17.4286 \dots \text{ at } (0.8406 \dots, -\infty, \infty)$$

(9) Kowalik and Osborne Function [68].

$$m = 4, n = 11$$

$$r_i(x) = y_i - \frac{x_1(u_i^2 + u_i x_2)}{u_i^2 + u_i x_3 + x_4}$$

where

i	$y_i$	$u_i$	i	$y_i$	$u_i$
1	0.1957	4.0000	7	0.0456	0.1250
2	0.1947	2.0000	8	0.0342	0.1000
3	0.1735	1.0000	9	0.0323	0.0833
4	0.1600	0.5000	10	0.0235	0.0714
5	0.0844	0.2500	11	0.0246	0.0625
6	0.0627	0.1670			

$$x_0 = (0.25, 0.39, 0.415, 0.39)$$

$$f = 3.07505 \dots 10^{-4}$$

$$f = 1.02734 \dots 10^{-3} \text{ at } (+\infty, -14.07 \dots, -\infty, -\infty)$$

(10) Meyer Function [79].

$$m = 3, n = 16$$

$$r_i(x) = x_1 \exp \left[ \frac{x_2}{45 + 5i + x_3} \right] - y_i$$

where

$i$	$y_i$	$i$	$y_i$
1	34780	9	8261
2	28610	10	7030
3	23650	11	6005
4	19630	12	5147
5	16370	13	4427
6	13720	14	3820
7	11540	15	3307
8	9744	16	2872

$x_0 = (0.02, 4000, 250)$

$$f = 87.9458 \dots$$

(11) Watson Function [68].

$$2 \leq m \leq 31, m = 31$$

$$r_i(x) = \sum_{j=2}^m (j-1)x_j \left(\frac{i}{29}\right)^{j-2} - \left(\sum_{j=1}^m x_j \left(\frac{i}{29}\right)^{j-1}\right) - 1, 1 \leq i \leq 29$$

$$r_{30}(x) = x_1$$

$$r_{31}(x) = x_2 - x_1^2 - 1$$

$$x_0 = (0, \dots, 0)$$

$$f = 2.28767 \dots \cdot 10^{-3} \text{ for } m = 6$$

$$f = 1.39976 \dots \cdot 10^{-6} \text{ for } m = 9$$

$$f = 4.72238 \cdots 10^{-10} \text{ for } m = 12$$

(12) Box Three-Dimensional Function [9].

$$m = 3, n \geq m$$

$$r_i(x) = \exp[-t_i x_1] - \exp[-t_i x_2] - x_3(\exp[-t_i] - \exp[-10t_i])$$

$$\text{where } t_i = 0.1i$$

$$x_0 = (0, 10, 20)$$

$$f = 0 \text{ at } (1, 10, 1), (10, 1, -1) \text{ and wherever } (x_1 = x_2 \text{ and } x_3 = 0)$$

(13) Jennrich and Sampson Function [65].

$$m = 2, n \geq m$$

$$r_i(x) = 2 + 2i - (\exp[ix_1] + \exp[ix_2])$$

$$x_0 = (0.30.4)$$

$$f = 124.362 \dots \text{ at } x_1 = x_2 = 0.2587 \dots \text{ for } n = 10$$

(14) Brown and Dennis Function [12].

$$m = 4, n \geq m$$

$$r_i(x) = (x_1 + t_i x_2 - \exp[t_i])^2 + (x_3 + x_4 \sin[t_i] - \cos[t_i])^2 \text{ where } t_i = i/5$$

$$x_0 = (25, 5, -5, -1)$$

$$f = 85822.2 \dots \text{ for } n = 20$$

(15) Chebyquad Function [37].

$$m \text{ is variable, } n \geq m$$

$$r_i(x) = \frac{1}{m} \sum_{j=1}^m T_i(x_j) - \int_0^1 T_i(x) dx \text{ where } T_i(x) \text{ is the } i^{\text{th}} \text{ Chebyshev}$$

polynomial shifted to the interval  $[0, 1]$

$$x_0 = (\xi_j) \text{ where } \xi_j = \frac{j}{m+1}$$

$$f = 0 \text{ for } m = n, 1 \leq n \leq 7, \text{ and } n = 9$$

$$f = 3.51687 \cdots 10^{-3} \text{ for } m = n = 8$$

$$f = 6.50395 \cdots 10^{-3} \text{ for } m = n = 10$$

(16) Brown Almost-Linear Function [11].

$m$  is variable,  $n = m$

$$r_i(x) = x_i + \sum_{j=1}^m x_j - (m+1), 1 \leq i < m$$

$$r_m(x) = \left( \prod_{j=1}^m x_j \right) - 1$$

$$x_0 = (0.5, \dots, 0.5)$$

$$f = 0 \text{ at } (\alpha, \dots, \alpha, \alpha^{m-1}) \text{ where } \alpha \text{ satisfies } m\alpha^m - (m+1)\alpha^{m-1} + 1 = 0;$$

in particular,  $\alpha = 1$

$$f = 1 \text{ at } (0, \dots, m+1)$$

(17) Osborne 1 Function [90].

$$m = 5, m = 33$$

$$r_i(x) = y_i - (x_1 + x_2 \exp[-t_i x_4] + x_3 \exp[-t_i x_5])$$

where  $t_i = 10(i-1)$  and

i	$y_i$	i	$y_i$	i	$y_i$
1	0.844	12	0.718	23	0.478
2	0.908	13	0.685	24	0.467
3	0.932	14	0.658	25	0.457
4	0.936	15	0.628	26	0.448
5	0.925	16	0.603	27	0.438
6	0.908	17	0.580	28	0.431
7	0.881	18	0.558	29	0.424
8	0.850	19	0.538	30	0.420
9	0.818	20	0.522	31	0.414
10	0.784	21	0.506	32	0.411
11	0.751	22	0.490	33	0.406

$x_0 = (0.5, 1.5, -1, 0.01, 0.02)$

$$f = 5.46489 \dots 10^{-5}$$

(18) Osborne 2 Function [90].

$$m = 11, m = 65$$

$$r_i(x) = y_i - (x_1 \exp[-t_i x_5] + x_2 \exp[-(t_i - x_9)^2 x_6] + x_3 \exp[-(t_i - x_{10})^2 x_7] + x_4 \exp[-(t_i - x_{11})^2 x_8])$$

where  $t_i = (i - 1)/10$  and

$i$	$y_i$	$i$	$y_i$	$i$	$y_i$
1	1.366	23	0.694	45	0.672
2	1.191	24	0.644	46	0.708
3	1.112	25	0.624	47	0.633
4	1.013	26	0.661	48	0.668
5	0.991	27	0.612	49	0.645
6	0.885	28	0.558	50	0.632
7	0.831	29	0.533	51	0.591
8	0.847	30	0.495	52	0.559
9	0.786	31	0.500	53	0.597
10	0.725	32	0.423	54	0.625
11	0.746	33	0.395	55	0.739
12	0.679	34	0.375	56	0.710
13	0.608	35	0.372	57	0.729
14	0.655	36	0.391	58	0.720
15	0.616	37	0.396	59	0.636

$i$	$y_i$	$i$	$y_i$	$i$	$y_i$
16	0.606	38	0.405	60	0.581
17	0.602	39	0.428	61	0.428
18	0.626	40	0.429	62	0.292
19	0.651	41	0.523	63	0.162
20	0.724	42	0.562	64	0.098
21	0.649	43	0.607	65	0.054
22	0.649	44	0.653		

$x_0 = (1.3, 0.65, 0.65, 0.7, 0.6, 3, 5, 7, 2, 4.5, 5.5)$   
 $f = 4.01377 \cdots 10^{-2}$

## References

- [1] H. Akaike. On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method. *Annals of the Institute of Statistical Mathematics*, 11:1–17, 1959.
- [2] Y. Bard. Comparison of gradient methods for the solution of nonlinear parameter estimation problems. *SIAM Journal on Numerical Analysis*, 7:157–186, 1970.
- [3] R. H. Barham and W. Drane. An algorithm for least squares estimation of nonlinear parameters when some of the parameters are linear. *Technometrics*, 14:757–766, 1972.
- [4] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148, 1988.
- [5] D. M. Bates and D. G. Watts. A relative offset orthogonality convergence criterion for nonlinear least squares. *Technometrics*, 23:179–183, 1981.
- [6] A. Björck. A direct method for sparse least squares problems with lower and upper bounds. *Numerische Mathematik*, 54:19–32, 1988.
- [7] I. Bongartz, A. R. Conn, N. Gould, and P. L. Toint. Constrained and unconstrained testing environment. *ACM Transactions on Mathematical Software*, 21:123–160, 1995.
- [8] J. F. Bonnans, E. R. Panier, A. L. Tits, and J. Zhou. Avoiding the maratos effect by means of a nonmonotone line search: II. inequality constrained problems - feasible iterates. *SIAM Journal on Numerical Analysis*, 29:1187–1202, 1992.
- [9] M. J. Box. A comparison of several current optimization methods, and the use of transformations in constrained problems. *The Computing Journal*, 9:67–77, 1966.
- [10] F. H. Branin and S. K. Hoo. A method for finding multiple extrema of a function of  $n$  variables. In F. A. Lootsma, editor, *Numerical Methods for Non-Linear Optimization*, pages 231–237. Academic Press, New York, 1972.

- [11] K. M. Brown. A quadratically convergent Newton-like method based upon Gaussian elimination. *SIAM Journal on Numerical Analysis*, 6:560–569, 1969.
- [12] K. M. Brown and J. E. Dennis. New computational algorithms for minimizing a sum of squares of nonlinear functions. Technical Report 71-6, Department of Computer Science, Yale University, New Haven, Connecticut, Mar 1971.
- [13] J. R. Bunch and C. P. Nielsen. Updating the singular value decomposition. *Numerische Mathematik*, 31:111–129, 1978.
- [14] R. M. Chamberlain, C. Lemarechal, H. C. Pedersen, and M. J. D. Powell. The watchdog technique for forcing convergence in algorithms for constrained optimization. *Mathematical Programming Studies*, 16:1–17, 1982.
- [15] S. N. Chow, J. Mallet-Paret, and J. A. Yorke. Finding zeros of maps: Homotopy methods that are constructive with probability one. *Mathematics of Computation*, 32:887–899, 1978.
- [16] M. T. Chu. On the continuous realization of iterative processes. *SIAM Review*, 30:375–387, 1988.
- [17] R. E. Cline. Note on the generalized inverse of the product of matrices. *SIAM Review*, 6:57–58, 1964.
- [18] R. E. Cline and R. J. Plemmons.  $l_2$ -solutions to underdetermined linear systems. *SIAM Review*, 18:92–106, 1976.
- [19] T. F. Coleman and A. R. Conn. Nonlinear programming via an exact penalty function: Asymptotic analysis. *Mathematical Programming*, 24:123–136, 1982.
- [20] T. F. Coleman and Y. Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal of Optimization*, 6:418–445, 1996.
- [21] A. R. Conn, N. I. M. Gould, and P. L. Toint. Correction to the paper on global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal of Numerical Analysis*, 26:764–767, 1988.

- [22] A. R. Conn, N. I. M. Gould, and P. L. Toint. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal of Numerical Analysis*, 25:433–460, 1988.
- [23] A. R. Conn, N. I. M. Gould, and P. L. Toint. Testing a class of methods for solving minimization problems with simple bounds on the variables. *Mathematics of Computation*, 50:399–430, 1988.
- [24] A. R. Conn, N. I. M. Gould, and P. L. Toint. Methods for nonlinear constraints in optimization calculations. Technical Report 96/6, Department of Mathematics, Facultés Universitaires ND de la Paix, Namur, Belgium, 1996.
- [25] N. de Villiers and D. Glasser. A continuation method for nonlinear regression. *SIAM Journal of Numerical Analysis*, 18:1139–1154, 1981.
- [26] N. Y. Deng, Y. Xiao, and F. J. Zhou. Nonmonotonic trust region algorithm. *Journal of Optimization Theory and Applications*, 76:259–285, 1993.
- [27] J. E. Dennis, D. M. Gay, and R. E. Welsch. An adaptive nonlinear least-squares algorithm. *ACM Transactions in Mathematical Software*, 7:348–368, 1981.
- [28] J. E. Dennis, D. M. Gay, and R. E. Welsch. Algorithm 573 NL2SOL - an adaptive nonlinear least-squares algorithm. *ACM Transactions in Mathematical Software*, 7:369–383, 1981.
- [29] J. E. Dennis, S.-B. Li, and R. A. Tapia. A unified approach to global convergence of trust region methods for nonsmooth optimization. *Mathematical Programming*, 68:319–346, 1995.
- [30] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1983.
- [31] J. E. Dennis and T. Steihaug. On the successive projections approach to least-squares problems. *SIAM Journal of Numerical Analysis*, 23:717–733, 1986.

- [32] L. Elden. Perturbation theory for the least squares problem with linear equality constraints. *SIAM Journal of Numerical Analysis*, 17:338–350, 1980.
- [33] J. Eriksson. *Optimization and Regularization of Nonlinear Least Squares Problems*. PhD thesis, Umeå University, Umeå Sweden, 1996.
- [34] F. Facchinei and S. Lucidi. Nonmonotone bundle-type scheme for convex nonsmooth minimization. *Journal of Optimization Theory and Applications*, 76:241–257, 1993.
- [35] M. C. Ferris, S. Lucidi, and M. Roma. Nonmonotone curvilinear line search methods for unconstrained optimization. *Computational Optimization and Applications*, 6:117–136, 1995.
- [36] M. C. Ferris and S. Ludici. Nonmonotone stabilization methods for nonlinear equations. *Journal of Optimization Theory and Applications*, 81:53–74, 1994.
- [37] R. Fletcher. Function minimization without evaluating derivatives – a review. *The Computing Journal*, 8:33–41, 1965.
- [38] R. Fletcher. A modified Marquardt subroutine for non-linear least squares. Technical Report AERE - R 6799, United Kingdom Atomic Energy Authority Research Group, 1971.
- [39] R. Fletcher. An algorithm for solving linearly constrained optimization problems. *Mathematical Programming*, 2:133–165, 1972.
- [40] R. Fletcher. Minimizing general functions subject to linear constraints. In F. A. Lootsma, editor, *Numerical Methods for Non-Linear Optimization*, pages 279–296. Academic Press, New York, 1972.
- [41] R. Fletcher. A model algorithm for the composite nondifferentiable optimization problem. *Mathematical Programming Studies*, 17:67–76, 1982.
- [42] R. Fletcher. An  $L_1$ -penalty method for nonlinear constraints. In P. Boggs, R. Byrd, and R. Schnabel, editors, *Numerical Optimization*, pages 26–40, Philadelphia, 1984. SIAM.
- [43] R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *The Computing Journal*, 6:163–168, 1963.

- [44] F. Freudenstein and B. Roth. Numerical solutions of systems of nonlinear equations. *Journal of the ACM*, 4:550–556, 1963.
- [45] W. Gander. Least squares with a quadratic constraint. *Numerische Mathematik*, 36:291–307, 1981.
- [46] D. M. Gay. Computing optimal locally constrained steps. *SIAM Journal of Scientific and Statistical Computing*, 2:186–197, 1981.
- [47] P. E. Gill and W. Murray. Algorithms for the solution of the nonlinear least-squares problem. *SIAM Journal Numerical Analysis*, 15:977–992, 1978.
- [48] G. H. Golub and C. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 1983.
- [49] G. H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal of Numerical Analysis*, 10:413–432, 1973.
- [50] G. H. Golub and U. von Matt. Quadratically constrained least squares and quadratic problems. *Numerische Mathematik*, 59:561–580, 1991.
- [51] H. J. Greenberg. Interpreting rate analysis. *SIGMAP Newsletter*, 17:33–37, 1974.
- [52] H. J. Greenberg. Computational testing: Why, how and how much. *ORSA Journal on Computing*, 2:94–97, 1990.
- [53] L. Grippo, F. Lampariello, and S. Lucidi. A nonmonotone line search technique for Newton’s method. *SIAM Journal of Numerical Analysis*, 23:707–716, 1986.
- [54] L. Grippo, F. Lampariello, and S. Lucidi. A truncated Newton method with nonmonotone line search for unconstrained optimization. *Journal of Optimization Theory and Applications*, 60:401–419, 1989.
- [55] L. Grippo, F. Lampariello, and S. Lucidi. A quasi-discrete Newton algorithm with a nonmonotone stabilization technique. *Journal of Optimization Theory and Applications*, 64:495–510, 1990.

- [56] L. Grippo, F. Lampariello, and S. Lucidi. A class of nonmonotone stabilization methods in unconstrained optimization. *Numerische Mathematik*, 59:779–805, 1991.
- [57] C. Gurwitz. A test for cancellation errors in quasi-Newton methods. *ACM Transactions on Mathematical Software*, 18:134–140, 1992.
- [58] M. El Hallabi and R. A. Tapia. A global convergence theory for arbitrary norm trust region methods for nonlinear equations. Technical Report 93-41, Department of Mathematical Sciences, Rice University, Houston, TX, 1993.
- [59] R. J. Hanson and F. T. Krogh. A quadratic-tensor model algorithm for nonlinear least-squares problems with linear constraints. *ACM Transactions on Mathematical Software*, 18:115–133, 1992.
- [60] M. D. Hebden. An algorithm for minimization using exact second derivatives. Technical Report T.P. 515, Atomic Energy Research Establishment, Harwell, England, 1973.
- [61] M. Heinkenschloss. Mesh independence for nonlinear least squares problems with norm constraints. *SIAM Journal of Optimization*, 3:81–117, 1993.
- [62] M. Heinkenschloss. On the solution of a two ball trust region subproblem. *Mathematical Programming*, 64:249–276, 1994.
- [63] K. L. Hiebert. An evaluation of mathematical software that solves nonlinear least squares problems. *ACM Transactions on Mathematical Software*, 7:1–16, 1981.
- [64] W. Hock and K. Schittkowski. Test examples for nonlinear programming codes. In *Lecture Notes in Economics and Mathematical Systems*, volume 187. Springer-Verlag, New York, 1981.
- [65] R. I. Jennrich and P. F. Sampson. Application of stepwise regression to nonlinear estimation. *Technometrics*, 10:63–72, 1968.
- [66] K. C. Kiwiel. Restricted step and Levenberg-Marquardt techniques in proximal bundle methods for nonconvex nondifferentiable optimization. *SIAM Journal of Optimization*, 6:227–249, 1996.

- [67] O. Knoth. A globalization scheme for the generalized Gauss-Newton method. *Numerische Mathematik*, 56:591–607, 1989.
- [68] J. S. Kowalik and M. R. Osborne. *Methods for Unconstrained Optimization Problems*. Elsevier North Holland, New York, 1969.
- [69] F. T. Krogh. Efficient implementation of a variable projection algorithm for nonlinear least squares. *Communications of the ACM*, 17:167–169, Errata pg. 591, 1974.
- [70] W. H. Lawton and E. A. Sylvestre. Elimination of linear parameters in nonlinear regression. *Technometrics*, 13:461–467, 1971.
- [71] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.
- [72] P. Lindström and P.Å. Wedin. A new linesearch algorithm for nonlinear least squares problems. *Mathematical Programming*, 29:268–296, 1984.
- [73] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley Publishing Company, Reading Massachusetts, 1973.
- [74] D. J. MacMillan. A new expanding subspace algorithm for nonlinear optimization. Master's thesis, Colorado School of Mines, 1985.
- [75] O. L. Mangasarian and M. V. Solodov. Backpropagation convergence via deterministic nonmonotone perturbed minimization. In G. Tesauro, J. D. Cowan, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 383–390. Morgan Kaufmann Publishers, San Francisco, CA, 1994.
- [76] N. Maratos. *Exact Penalty Function Algorithms for Finite Dimensional and Optimization Problems*. PhD thesis, Imperial College of Science and Technology, London, U.K., 1978.
- [77] D. Marquardt. An algorithm for least-squares estimation of non-linear parameters. *SIAM Journal of Applied Mathematics*, 11:431–441, 1963.
- [78] D. Q. Mayne and E. Polak. A superlinearly convergent algorithm for constrained optimization problems. *Mathematical Programming Studies*, 16:45–61, 1982.

- [79] R. R. Meyer. Theoretical and computational aspects of nonlinear regression. In J. B. Rowen, O. L. Mangasarian, and K. Ritter, editors, *Nonlinear Programming*, pages 465–486. Academic Press, New York, 1970.
- [80] J. J. Moré. The Levenberg-Marquardt algorithm: Implementation and theory. In G. A. Watson, editor, *Proceedings of the Dundee Conference on Numerical Analysis*, volume 630 of *Lecture Notes in Mathematics*, pages 105–116. Springer-Verlag, 1978.
- [81] J. J. Moré. Recent developments in algorithms and software for trust region methods. In A. Bachern, M. Grottschel, and B. Korte, editors, *Mathematical Programming: The State of the Art*, pages 258–287. Springer-Verlag, Berlin, 1983.
- [82] J. J. Moré, B. S. Garbow, and K. E. Hillstom. User guide for minpack-1. Technical Report ANL-80-74, Argonne National Labs, 1980.
- [83] J. J. Moré, B. S. Garbow, and K. E. Hillstom. Fortran subroutines for testing unconstrained optimization software. *ACM Transactions in Mathematical Software*, 7:136–140, 1981.
- [84] J. J. Moré, B. S. Garbow, and K. E. Hillstom. Testing unconstrained optimization software. *ACM Transactions in Mathematical Software*, 7:17–41, 1981.
- [85] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM Journal of Scientific and Statistical Computing*, 4:553–572, 1983.
- [86] J. J. Moré and G. Toraldo. Algorithms for bound constrained quadratic programming problems. *Numerische Mathematik*, 55:377–400, 1989.
- [87] L. Nazareth. Some recent approaches to solving large residual nonlinear least squares problems. *SIAM Review*, 22:1–11, 1980.
- [88] J. Nocedal. Theory of algorithms for unconstrained optimization. In *Acta Numerica 1992*, pages 199–242. Cambridge University Press, 1992.
- [89] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

- [90] M. R. Osborne. Some aspects of nonlinear least squares calculations. In F. A. Lootsma, editor, *Numerical Methods for Non-Linear Optimization*, pages 171–189. Academic Press, New York, 1972.
- [91] E. R. Panier and A. L. Tits. Avoiding the maratos effect by means of a nonmonotone line search: I. general constrained problems. *SIAM Journal on Numerical Analysis*, 28:1183–1195, 1991.
- [92] M. J. D. Powell. An iterative method for finding stationary values of a function of several variables. *The Computing Journal*, 5:147–151, 1962.
- [93] M. J. D. Powell. A Fortran subroutine for unconstrained minimization, requiring first derivatives of the objective function. Technical Report AERE - R 6469, United Kingdom Atomic Energy Authority Research Group, 1970.
- [94] M. J. D. Powell. A hybrid method for non-linear equations. *Numerical Methods for Non-Linear Algebraic Equations*, pages 87–114, 1970.
- [95] M. J. D. Powell. A new algorithm for unconstrained optimization. In J. B. Rosen, O. L. Mangasarian, and K. Ritter, editors, *Nonlinear Programming*, pages 31–65. Academic Press, New York, 1970.
- [96] M. J. D. Powell. On the global convergence of trust region algorithms for unconstrained minimization. *Mathematical Programming*, 29:297–303, 1984.
- [97] M. J. D. Powell. Convergence properties of algorithms for nonlinear optimization. *SIAM Review*, 28:487–500, 1986.
- [98] H. Ramison and P. Å. Wedin. A comparison of some algorithms for the nonlinear least squares problem. *Nordisk Tidskrift Informationsbehandling (BIT)*, 17:72–90, 1977.
- [99] C. R. Rao and S. K. Mitra. *Generalized Inverse of Matrices and its Applications*. John Wiley and Sons, Inc., New York, 1971.
- [100] S. M. Robinson. Quadratic interpolation is risky. *SIAM Journal of Numerical Analysis*, 16:377–379, 1979.
- [101] R. T. Rockafellar. Lagrange multipliers and optimality. *SIAM Review*, 35:183–238, 1993.

- [102] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computing Journal*, 3:175–184, 1960.
- [103] A. Ruhe. Accelerated Gauss-Newton algorithms for nonlinear least-squares problems. *Nordisk Tidskrift Informationsbehandling (BIT)*, 19:356–367, 1979.
- [104] D. E. Salane. A continuation approach for solving large-residual nonlinear least squares problems. *SIAM Journal of Scientific and Statistical Computing*, 8:655–671, 1987.
- [105] R. Schaback. Convergence analysis of the general Gauss-Newton algorithm. *Numerische Mathematik*, 46:281–309, 1985.
- [106] K. Schittkowski. Nonlinear programming codes. In *Lecture Notes in Economics and Mathematical Systems*, volume 183. Springer-Verlag, Berlin, 1980.
- [107] K. Schittkowski. The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function part 1: Convergence analysis. *Numerische Mathematik*, 38:83–114, 1981.
- [108] K. Schittkowski. The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function part 2: An efficient implementation with linear least squares subproblems. *Numerische Mathematik*, 38:115–127, 1981.
- [109] G. Schuller. On the order of convergence of certain quasi-Newton-methods. *Numerische Mathematik*, 23:181–192, 1974.
- [110] G. A. Shultz, R. B. Schnabel, and R. H. Byrd. A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties. *SIAM Journal of Numerical Analysis*, 22:47–67, 1985.
- [111] M. V. Solodov. *Nonmonotone and Perturbed Optimization*. PhD thesis, University of Wisconsin, Madison, Wisconsin, 1995.
- [112] D. C. Sorensen. Newton’s method with a model trust region modification. *SIAM Journal of Numerical Analysis*, 19:409–426, 1982.

- [113] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal of Numerical Analysis*, 20:626–637, 1983.
- [114] G. W. Stewart. On the perturbation of pseudo-inverses, projections, and linear least squares problems. *SIAM Review*, 19:634–662, 1977.
- [115] P. L. Toint. An assessment of non-monotone linesearch techniques for unconstrained optimization. Technical Report 94/14, Department of Mathematics, Facultés Universitaires ND de la Paix, Namur, Belgium, 1994.
- [116] P. L. Toint. An assessment of non-monotone linesearch techniques for unconstrained optimization: The complete numerical results. Technical Report 94/15, Department of Mathematics, Facultés Universitaires ND de la Paix, Namur, Belgium, 1994.
- [117] P. L. Toint. A non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints. Technical Report 94/24, Department of Mathematics, Facultés Universitaires ND de la Paix, Namur, Belgium, 1994.
- [118] P. L. Toint. A non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints: The complete numerical results. Technical Report 94/26, Department of Mathematics, Facultés Universitaires ND de la Paix, Namur, Belgium, 1994.
- [119] P. L. Toint. An assessment of nonmonotone linesearch techniques for unconstrained optimization. *SIAM Journal of Scientific Computing*, 17:725–739, 1996.
- [120] P. L. Toint. A non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints. *Mathematical Programming*, 77:69–94, 1997.
- [121] A. van der Sluis. Stability of the solutions of linear least squares problems. *Numerische Mathematik*, 23:241–254, 1975.
- [122] L. T. Watson. Numerical linear algebra aspects of globally convergent homotopy methods. *SIAM Review*, 28:529–545, 1986.
- [123] M. A. Wolfe. Extended iterative methods for the solution of operator equations. *Numerische Mathematik*, 31:153–174, 1978.

- [124] P. Wolfe. Convergence conditions for ascent methods. *SIAM Review*, 11:226–235, 1969.
- [125] P. Wolfe. Convergence conditions for ascent methods II: Some corrections. *SIAM Review*, 13:185–188, 1971.
- [126] Y. Xiao and F. J. Zhou. Nonmonotone trust region methods with curvilinear path in unconstrained minimization. *Computing*, 48:303–317, 1992.
- [127] Y. Yuan. Global convergence of trust region algorithms for nonsmooth optimization. Technical Report DAMTP 1983/NA13, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge, England, 1983.
- [128] W. I. Zangwill. *Nonlinear Programming: A Unified Approach*. Prentice-Hall, Englewood Cliffs, New Jersey, 1969.
- [129] J. Zhou and A. L. Tits. User’s guide for FSQP version 2.0 - a Fortran code for solving optimization problems, possibly minimax, with general inequality constraints and linear equality constraints, generating feasible iterates. Technical Report SRC TR-90-60, Systems Research Center, University of Maryland, College Park, MD, 1990.
- [130] J. L. Zhou and A. L. Tits. Nonmonotone line search for minimax problems. *Journal of Optimization Theory and Applications*, 76:455–476, 1993.