ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

DEPARTMENT OF MATHEMATICS

A Graduate Seminar Report
On
Globally Convergent Modifications of Newton's Method

By

Esubalew Alemayehu

Advisor:
Prof.Dr.rer.nat.habil.R.Deumlich

June 2005
Addis Ababa

Globally Convergent Modifications

Of

Newton's Method

Acknowledgement

First of all, I thank the almighty God, with the help of Whom this seminar report took its present form.

I would like to express my heart felt gratitude to my advisor and instructor, Prof.Dr.rer.nat.habil.R.Deumlich not only for his genuine advice, constructive comments, invaluable suggestions and provision of materials in preparing this seminar report but also in other social aspects showing me the bright future.

Finally, I would like to thank my sisters Aster Wondim and Melekam Denekew, for their over all responsibility of any social affairs, and continued encouragement to complete the study.

Contents page

Introduction i

1. NEWTON'S METHOD 1

1.1 The Newton's Method. 1

1.2 The Fast Local Convergence of Newton's Method. 2

1.2.1 Rate of Convergence. 2

1.3 The Quasi-Newton Frame Work. 5

1.4 Descent Directions. 6

2. Line Searches 9

2.1 The Line Search Approach. 9

2.2 Convergence Results for Properly Chosen Steps. 13

2.3 Step Selection by Backtracking. 19

2.3.1 Strategy for Reducing λ_k (choosing ρ). 20

3. Model-Trust Region 24

3.1 The Model -Trust Region Approach. 24

3.1.1 Selection of a Step Maximal Length σ_c from x_c. 24

3.2 Model -Trust Region Algorithm. 27

3.3 Computational Methods to Determine μ_c. 28

3.3.1 The Locally Constrained Optimal ("Hook") Step. 28

3.3.2 The Double Dogleg Step. 32

3.4 Up Dating the Trust Region. 38

3.5 Global Method for Systems of Non-linear Equations. 39

References 44

0. INTRODUCTION

It is known that numerical methods play a vital role in solving a broad and important class of optimization problems. Almost all numerical methods for determining the optimum of a given non-linear objective function are iterative and starting from a given initial estimate for the solution they proceed by generating a sequence of new estimates, each of which represents an improvement over the previous ones.

The most effective numerical methods are based upon some variant of Newton's method for finding a zero of the gradient of the objective function. But Newton's method must be modified in various ways in order to produce a robust optimization algorithm.

Newton's method was shown to be locally q-*quadratically* convergent. This means that when the current solution approximation is good enough, it will be improved rapidly and with relative ease. Unfortunately, it is not unusual to expend significant computational effort in getting close enough. In addition, the strategies for getting close constitute the major part of the program and the programming effort, and they can be sensitive to small difference in implementation. Therefore, it is necessary to modify the Newton's method so as it can converge to the solution, from any starting point.

This paper will be devoted to the two major approaches proceeding from a solution estimate out side the convergence region of Newton's method. The two major approaches are *line- searches* approach and *model-trust region* approach.

The remainder of this paper contains 3 chapters. In chapter 1, we introduce the Newton's method designed for systems of non linear equations, and will set the framework for the class of algorithm we want to consider. Chapter 2 discusses the first major global approach, modern versions of the traditional idea of back tracking along the Newton direction if a full Newton step is unsatisfactory. Chapter 3 discusses the second major approach, based on estimating the region in the local model, underlying Newton's method, can be trusted to adequately represent the function, and taking a step to approximately minimize the model in this region. Both

approaches are derived initially for unconstrained minimization problem; their application to solving systems of non linear equations is the topic of the last section of Chapter 3.

We will denote the solution by x_*. Usually we will be speaking of a single iteration, and will denote the current iterate by x_c and the new iterate by x_+. The first iterate will be denoted by x_0, the sequence of iterates by $\{x_k\}$, the vector of first partial derivative of f at x by $\nabla f(x)$, the matrix of second partial derivatives of f at x, i.e. the Hessian Matrix, by $\nabla^2 f(x)$, and the end of a proof by //.

1. NEWTON'S METHOD

This chapter begins our consideration of multivariable problems by deriving and discussing Newton's Method for systems of nonlinear equations, the quickly convergent local method that will be the basic model for our algorithms.

1.1 THE NEWTON'S METHOD

Newton's method is usually used to determine the solution of a system of non-linear equations:

$$\text{given } F:\mathbb{R}^n \to \mathbb{R}^n, \quad \text{find } x_* \in \mathbb{R}^n \text{ such that } F(x_*) = 0 \qquad (1.1.1)$$

where F is assumed to be continuously differentiable.

In this section we derive Newton's method for problem (1.1.1) and discuss its local convergence characteristics.

Newton's Method for problem (1.1.1) is derived by finding the root of an affine approximation to F at the current iterate x_c which comes simply and naturally from Newton's Theorem for multidimensional case, that is

$$F(x_c + p) = F(x_c) + \int_{x_c}^{x_c+p} J(z)dz, \qquad (1.1.2)$$

where $J(z)$ is the Jacobian matrix of F at z.

Therefore, it seems reasonable to approximate the indefinite integral by

$$\int_{x_c}^{x_c+p} J(z)dz \approx J(x_c)\cdot p \qquad (1.1.3)$$

and get the affine approximation to F at perturbation p of x_c,

$$M_c(x_c + p) = F(x_c) + J(x_c)p. \qquad (M_c \text{ stands for current model}) \qquad (1.1.4)$$

Next we solve for the step s^N that makes $M_c(x_c + s^N) = 0$, giving the Newton iteration for (1.1.1). Solve,

$$J(x_c)s^N = -F(x_c)$$
$$x_+ = x_c + s^N. \qquad (1.1.5)$$

(An equivalent way to view this procedure is that we are finding a simultaneous zero of the affine models of the n-component functions of F given by

$$(M_c)_i(x_c + s^N) = f_i(x_c) + \nabla f_i(x_c)^T \cdot s^N, \quad i = 1,...,n \qquad (1.1.6))$$

Since x_+ is not expressed to be the exact solution to (1.1.1) but only to be a better estimate than x_c, we make the Newton iteration (1.1.4) into an algorithm by applying it iteratively from a starting guess x_0.

We have the following:

ALGORITHM 1.1.1: NEWTON'S METHOD FOR SYSTEMS OF NON LINEAR EQUATIONS

Given F:$\mathrm{R}^n \rightarrow \mathrm{R}^n$ continuously differentiable and $x_0 \in \mathrm{R}^n$:

Then at each iteration k, solve

$$J(x_k)s_k = -F(x_k),$$
$$x_{k+1} = x_k + s_k. \qquad (1.1.7)$$

1.2 THE FAST LOCAL CONVERGENCE OF NEWTON'S METHOD

1.2.1 RATE OF CONVERGENCE

Given an iterative method that produces a sequence of points $x_1, x_2, x_3....$

From a starting guess x_0, we want to know, if the iterates converge to a solution x_*, and if so, how quickly. If we assume that we know what it means to write

$$\lim_{k\to\infty} a_k = 0 \qquad (1.2.1)$$

for a real sequence $\{a_k\}$, then the following definition characterizes the properties we will need.

Definition 2.2.1: Let $x_* \in \mathbb{R}$, $x_k \in \mathbb{R}$, k=0,1,2,...then the sequence

$\{x_k\}=\{x_0, x_1, x_2, x_3....\}$ is said to converge to x_* if

$$\lim_{k\to\infty}|x_k - x_*| = 0. \qquad (1.2.2)$$

If in addition, there exists a constant $c \in [0,1)$ and an integer $\hat{k} \geq 0$ such that

for all $k \geq \hat{k}$,

$$|x_{k+1} - x_*| \leq c|x_k - x_*| \qquad (1.2.3)$$

then $\{x_k\}$ is said to be *q-linearly convergent* to x_*.

If for some sequence $\{c_k\}$ that converges to 0,

$$|x_{k+1} - x_*| \leq c_k|x_k - x_*|\ , \qquad (1.2.4)$$

then $\{x_k\}$ is said to converge *q-super linearly* to x_*.

If there exist constants p >1, $c \geq 0$ and $\hat{k} \geq 0$ such that $\{x_k\}$ converges to x_*

and for all $k \geq \hat{k}$,

$$|x_{k+1} - x_*| \leq c|x_k - x_*|^p, \qquad (1.2.5)$$

then $\{x_k\}$ is said to converge to x_* with $q-$*order at least p*.

If p =2 or 3, the convergence is said to be *q-quadratic* or *cubic*, respectively.

In practice, q-*linear* convergence can be fairly slow, where as q-*quadratic* or q-*super linear* convergence is eventually quite fast.

The main advantage of Newton's method is that, if x_0 is close enough to a solution x_* and $J(x_*)$ is non singular, the elements x_k generated by Algorithm 1.1.1 converge q-*quadratic ally* to x_* .

For example, let

$$F(x) = \begin{bmatrix} x_1 + x_2 - 3 \\ x_1^2 + x_2^2 - 9 \end{bmatrix}$$

which has roots at $(3,0)^{\mathrm{T}}$ and $(0,3)^{\mathrm{T}}$, and let $x_0 = (1,5)^T$. Then the first two iterations of Newton's method are

$$J(x_0)s_0 = -F(x_0),$$

$$\begin{bmatrix} 1 & 1 \\ 2 & 10 \end{bmatrix} s_0 = -\begin{bmatrix} 3 \\ 17 \end{bmatrix}, \quad s_0 = \begin{bmatrix} -\frac{13}{8} \\ -\frac{11}{8} \end{bmatrix},$$

$$x_1 = x_0 + s_0 \approx (-0.625, 3.625)^T,$$

$$J(x_1)s_1 = -F(x_1),$$

$$\begin{bmatrix} 1 & 1 \\ -\frac{5}{4} & \frac{29}{4} \end{bmatrix} s_1 = \begin{bmatrix} 0 \\ \frac{145}{32} \end{bmatrix}, \quad s_1 = \begin{bmatrix} \frac{145}{272} \\ -\frac{145}{272} \end{bmatrix}$$

$$x_2 = x_1 + s_1 \approx (-0.092, 3.092)^T.$$

Newton's method seems to be working well here; x_2 is already quite close to the root (0, 3) $^{\mathrm{T}}$. On the other hand, Newton's Method naturally will be no better at converging from bad starting guesses for multi variable nonlinear problems.

For example, if

$$F(x) = \begin{bmatrix} e^{x_1} - 1 \\ e^{x_2} - 1 \end{bmatrix},$$

$x_* = (0,0)^T$, and $x_0 = (-10,-10)^T$, then

$$x_1 = (-11 + e^{10}, -11 + e^{10})^T$$

$$\approx (2.2 \times 10^4, 2.2 \times 10^4)^T,$$

which is not a very good step! Therefore the convergence characteristics of Newton's Method indicate how to use it in multidimensional algorithms: we will always want to use it in at least the final iterations of any non linear algorithms to take advantage of its fast local convergence, but it will have to be modified in order to converge the solution, from any starting point. Such type of algorithm is said to be *globally convergent algorithm* or *global method*, which is the task of this paper.

1.3 THE QUASI-NEWTON FRAME WORK

The basic idea in forming a successful non linear algorithm is to combine a globally convergent strategy with a fast local strategy in a way that derives the benefits of both. The frame work for doing this is out lined in algorithm below. The most important point is to try Newton's Method, or some modification of it, first at each iteration. If it seems to be taking a reasonable step –for example, if f decreases in a minimization application–use it. If not fall back on a step dictated by a global method. Such a strategy will always end up using Newton's Method close to the solution and thus retain its fast local convergence rate. If the global method is chosen and incorporated properly; the algorithm will also be globally convergent. We will call an algorithm that takes this approach Quasi-Newton.

ALGORITHM 1.3.1: QUASI-NEWTON ALGORITHM FOR NONLINEAR EQUATIONS OR UNCONSTRAINED OPTIMIZATION [FOR OPTIMIZATION, REPLACE $F(x_k)$ BY $\nabla f(x_k)$ AND J_k BY $H_k = \nabla^2 f(x_k)$].

Given $F:\mathrm{R}^n \rightarrow \mathrm{R}^n$ continuously differentiable, and $x_0 \in \mathrm{R}^n$.

At each iteration k we will do:

1. Compute $F(x_k)$, if it is not already done, and decide whether to stop or continue.
2. Compute J_k to be $J(x_k)$, or an approximation to it.
3. Apply a factorization technique to J_k and estimate its condition number. If J_k is ill- conditioned, perturb it in an appropriate manner.
4. Solve $J_k s_k^N = -F(x_k)$
5. Decide whether to take a Newton step, $x_{k+1} = x_k + s_k^N$, or to choose x_{k+1} by a global strategy. This step often furnishes $F(x_k)$ to step1.

1.4 DESCENT DIRECTIONS

We will start our discussion of global methods by considering the unconstrained minimization problem.

$$\min f: \mathrm{R}^n \rightarrow \mathrm{R}, \qquad x \in \mathrm{R}^n \ , \tag{1.4.1}$$

because there is a natural global strategy for minimization problems: it is to make sure that each step decreases the value of f . There are corresponding strategies for systems of non linear equations, in particular, making sure each step decreases the value of some norm of $F:\mathrm{R}^n \rightarrow \mathrm{R}^n$. However, by using a norm, one is really transforming the problem of solving the systems of equations $F(x) = 0$ in to the problem of minimizing a function, $f:\mathrm{R}^n \rightarrow \mathrm{R}$ where

$$f := \tfrac{1}{2}\|F\|_2^2 \tag{1.4.2}$$

Therefore, this paper will discuss global strategies for unconstrained minimization. In the last section of chapter 3 we will apply these strategies to systems of non linear equations, using the transformation (1.4.2).

The basic idea of a global method for unconstrained minimization is geometrical obvious: take steps that lead "down hill" for the function f. More precisely, one chooses a direction p from the current point x_c in which f decreases initially, and a new point x_+ in this direction from x_c such that $f(x_+) < f(x_c)$. Such a direction is called a *descent direction*. Mathematically, p is a descent direction from x_c, if the directional derivative of f at x_c in the direction p is negative, i.e. if

$$\nabla f(x_c)^T p < 0. \tag{1.4.3}$$

Descent direction forms the basis of some global methods for minimization and are important to all of them, and therefore they are the topic of this section. The only direction we will consider for minimization is the Newton direction $s^N = -H_c^{-1}\nabla f(x_c)$, where H_c is either $\nabla^2 f(x_c)$ or an approximation of it. Therefore, it is natural to ask, whether the Newton direction is a descent direction. By (1.4.3) the direction s^N is a descent direction if and only if

$$\nabla f(x_c)^T s^N = -\nabla f(x_c) H_c^{-1} \nabla f(x_c) < 0,$$

which is true if H_c^{-1} or equivalently, H_c is positive definite This guarantees not only a descent direction but also a quadratic model with a unique minimizer. Furthermore, if the Newton direction is a descent direction, it is guaranteed that for sufficiently small positive δ,

$$f(x_c + \delta p) < f(x_c),$$

That is, the function value will be decreased. Therefore, in all our methods for unconstrained minimization, the model Hessian H_c will be formed or coerced to make it positive definite.

A second natural question is the following. As long as one is taking steps in descent directions, what is the direction p in which f decreases most rapidly from x? Since, for a given perturbation vector p, the directional derivative of f in the

direction p is directly proportional to the length of p, we can pose our question about a most rapid descent direction for a given norm as

$$\min_{p \in R^n} \nabla f(x)^T p \quad \text{subject to} \|p\| = 1,$$

which, in the l_2 norm, has solution $p = \frac{-\nabla f(x)}{\|\nabla f(x)\|_2}$. Thus the negative of the gradient direction is the steepest downhill direction from x in the l_2 norm, and it is referred to as the steepest -descent direction.

A classical minimization algorithm due to Cauchy is based solely on the steepest-descent direction. But it is not a computational method, because each step contains an exact one dimensional minimization problem and the convergence is only linear. However, when our global strategy must take step much smaller than the Newton step, we will see that it may be take steps in or close to, the steepest descent direction.

2. LINE SEARCHES

2.1 THE LINE SEARCHE APPROACH

The first strategy we consider for proceeding from a solution estimate outside the convergence region of Newton's method is the method of line searches. The idea of a line search algorithm is the following:

ALGORITHM 2.1.1: Given a descent direction p_k, we take a step in that direction that yields an "acceptable " x_{k+1} .i.e,

at iteration k:

Calculate a descent direction p_k,

Set $x_{k+1} := x_k + \lambda_k p_k$ for some $\lambda_k > 0$ that makes

x_{k+1} an acceptable next iterate.

How ever, rather than setting p_k to the steepest descent direction $-\nabla f(x_k)$, we will use the Quasi-Newton's direction $-H_k^{-1}\nabla f(x_k)$, where $H_k = \nabla^2 f(x_k) + \mu_k I$ is positive definite with $\mu_k = 0$, if $\nabla^2 f(x_k)$ is safely positive definite. This will allows us to retain fast local convergence.

The term "line search" refers to a procedure for choosing λ_k in the above Algorithm 2.1.1. The common procedure now is to try for $\lambda_k = 1$(the full quasi-Newton step) first and, if λ_k=1 fails to satisfy the criterion in use, to back track in a systematic way along the direction defined by that step. Computational experience has shown the importance of taking a full quasi-Newton step whenever possible. Failure to do so leads to forfeiture of the advantage of Newton's method near the solution. Therefore, it is important that any procedure for choosing λ_k allow λ_k=1 near the solution. Since there is no compelling reason to confine our discussion to line

search in the quasi-Newton's direction, we consider the search for $x_+ = x_c + \lambda_c p$ a long a descent direction p from the current solution estimate x_c.

While no step–acceptance rule will always be optimal, it does seem to be common sense to require that

$$f(x_{k+1}) < f(x_k) \qquad (2.1.1)$$

but this simple condition dose not guarantee that $\{x_k\}$ will converge to a minimizer of f.

If very small reductions in f are taken relative to the lengths of the steps, then any limiting value of $\{f(x_{k+1})\}$ may not locally minimizer.

For example, let $f(x) = x^2$, $x_0 = 2$. If we choose $\{p_k\} = \{(-1)^{k+1}\}, \{\lambda_k\} = \{2+3(2^{-(k+1)})\}$, then $\{x_k\} = \{2, \frac{-3}{2}, \frac{5}{4}, \frac{-9}{8}, ,,,\} = \{(-1)^k (1+2^{-k})\}$, each p_k is a descent direction from x_k, and $f(x_k)$ is monotonically decreasing with $\lim_{k\to\infty} f(x_k) = 1$. Of course this is not a minimum of any sort for f, and furthermore $\{x_k\}$ has limit points ± 1, so it does not converge.

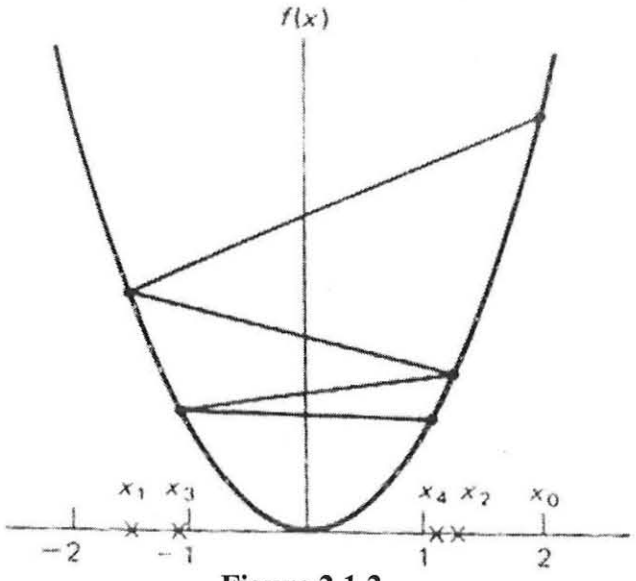

Figure 2.1.2

In Figure 2.1.2 we show the monotonically decreasing sequences of iterates that don't converge to the minimizer.

We can fix this by requiring that the average rate of decrease from $f(x_c)$ to $f(x_+)$ be at least some prescribed fraction of the initial rate decrease in that direction; that is we pick an $\alpha \in (0,1)$ and choose λ_c from among those $\lambda > 0$ that satisfy

$$f(x_c + \lambda p) \le f(x_c) + \alpha\lambda \nabla f(x_c)^T p \ .$$

Equivalently, $f(x_+) \le f(x_c) + \alpha \nabla f(x_c)^T (x_+ - x_c)$. (2.1.2)

It is possible also that if the steps are too small, relative to the initial rate of decrease of f, then any limiting value of $\{x_k\}$ may not be a local minimizer of f.

For example consider the same function with the same initial estimate ,and let us take $\{p_k\} = \{-1\}, \{\lambda_k\} = \{2^{-k+1}\}$, then $\{x_k\} = \{2, \frac{3}{2}, \frac{5}{4}, \frac{9}{8}, ,,,\} = \{1+2^{-k}\}$, each p_k is again a descent direction , $f(x_k)$ decreases monotonically, and $\lim_{k\to\infty} x_k = 1$, which is not a minimizer of f.

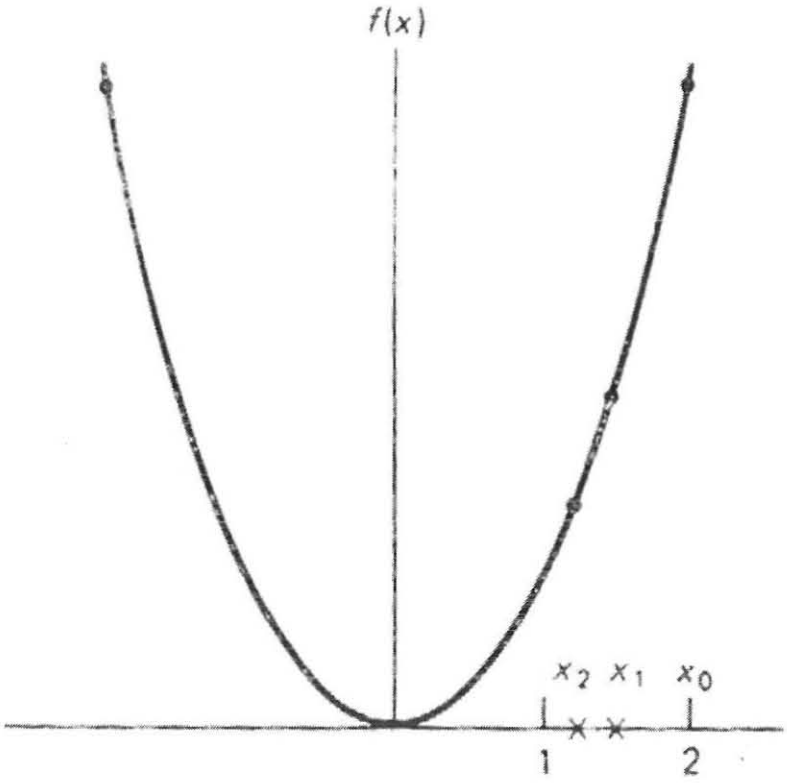

Figure 2.1.3

In Figure 2.1.3 we can see also that monotonically decreasing sequences of iterates that don't converge to the minimizer.

To ensure sufficiently large steps; we will require that the rate of decrease of f in the direction p at x_+ be larger than some prescribed fraction of the rate of decrease in the direction p at x_c; that is,

$$\nabla f(x_k)^T p \ \nabla f(x_c + \lambda_c p)^T p \geq \beta \nabla f(x_c)^T p . \qquad (2.1.3)$$

Equivalently,

$$\nabla f(x_+)^T (x_+ - x_c) \geq \beta \nabla f(x_c)^T (x_+ - x_c), \text{ for some fixed constant } \beta \in (\alpha,1).$$

In practice, (2.1.3) generally is not needed because the use of a back tracking strategy avoids excessively small steps.

2.2. CONVERGENCE RESULTS FOR PROPERLY CHOSEN STEPS

The following theorem is due to Wolfe. It shows that given any descent direction p_k, there exist $x_k + \lambda_k p_k$ satisfying (2.1.2) and (2.1.3).

Theorem 2.2.1: let $f: \mathrm{R}^n \to \mathrm{R}$ be continuously differentiable on R^n. let $x_k \in \mathrm{R}^n$, $p_k \in \mathrm{R}^n$ obey $\nabla f(x_k)^T p_k < 0$, and assume $\{f(x_k + \lambda p_k) / \lambda > 0\}$ is bounded below. Then if $0 < \alpha < \beta < 1$, there exist $\lambda_u > \lambda_l > 0$ such that $x_{k+1} = x_k + \lambda_k p_k$ satisfies (2.1.2) and (2.1.3), if $\lambda_k \in (\lambda_l, \lambda_u)$.

Proof. Since $\alpha < 1, f(x_k + \lambda p_k) < f(x_k) + \alpha\lambda\nabla f(x_k)^T p_k$ for all $\lambda > 0$ sufficiently small. Since $f(x_k + \lambda p_k)$ is also bounded below, there exist some smallest positive $\hat{\lambda}$ such that

$$f(\hat{x}) = f(x_k + \hat{\lambda} p_k) = f(x_k) + \alpha \hat{\lambda} \nabla f(x_k)^T p_k . \qquad (2.2.1)$$

Thus any $x \in (x_k, \hat{x})$ satisfies (2.1.2). By mean value theorem, there exist $\hat{\lambda} \in (0, \hat{\lambda})$ such that

$$\begin{aligned} f(\hat{x}) - f(x_k) &= \nabla f(x_k + \bar{\lambda} p_k)^T (\hat{x} - x_k) \\ &= \nabla f(x_k + \bar{\lambda} p_k)^T \hat{\lambda} p_k \end{aligned} \qquad (2.2.2)$$

and so from (2.2.1) and (2.2.2),

$$\nabla f(x_k + \bar{\lambda} p_k)^T p_k = \alpha \nabla f(x_k)^T p_k > \beta \nabla f(x_k)^T p_k \qquad (2.2.3)$$

since $\alpha < \beta$ and $\nabla f(x_k)^T p_k < 0$. By the continuity of ∇f, (2.2.3) still holds for λ in some interval (λ_l, λ_u) about $\bar{\lambda}$. Therefore, if we restrict (λ_l, λ_u) to be in $(0, \hat{\lambda}), x_{k+1} = x_k + \lambda_k p_k$ satisfies (2.1.2) and (2.1.3) for any $\lambda_k \in (\lambda_l, \lambda_u)$. //

Theorem 2.2.2: let $f:\mathbb{R}^n \to \mathbb{R}$ be continuously differentiable on $\mathbb{R}^n$, and assume there exists $\gamma \geq 0$ such that

$$\|\nabla f(z) - \nabla f(x)\|_2 \leq \gamma \|z - x\|_2 \tag{2.2.3}$$

for every x, $z \in \mathbb{R}^n$. Then, given any $x_0 \in \mathbb{R}^n$, either f is unbounded below, or there exists a sequence $\{x_k\}$, $k = 0,1,\ldots$, obeying (2.1.2), (2.1.3) and either

$$\nabla f(x_k)^T s_k < 0 \text{ or } \nabla f(x_k) = 0 \tag{2.2.4}$$

and $s_k = 0$, for each $k \geq 0$, where

$$s_k := x_{k+1} - x_k$$

Furthermore, for any such sequence, either

(a) $\nabla f(x_k) = 0$ for some $k \geq 0$, or

(b) $\lim_{k\to\infty} f(x_k) = -\infty$, or

(c) $\lim_{k\to\infty} \dfrac{\nabla f(x_k)^T s_k}{\|s_k\|_2} = 0$

Proof. For each k, if $\nabla f(x_k) = 0$, then (a) holds and the sequence is constant subsequently. If $\nabla f(x_k) \neq 0$, then there exists p_k, e.g. $p_k = -\nabla f(x_k)$ such that $\nabla f(x_k)^T p_k < 0$. By Theorem 2.2.1 either f is unbounded below, or there exists $\lambda_k > 0$ such that $x_{k+1} = x_k + \lambda_k p_k$ satisfies (2.1.2) and (2.1.3). In order to simplify notation, let us assume that $\|p_k\|_2 = 1$, so $\lambda_k = \|s_k\|_2$. This constitutes no loss of generality.

So far we see that either f is unbounded below, or $\{x_k\}$ exists, and either $\{x_k\}$ satisfies (a) or else $s_k \neq 0$ for every k. We must now show that if no term of $\{s_k\}$ is zero, then (b) or (c) must hold. It will be useful to have

$$\sigma_k := \frac{\nabla f(x_k)^T s_k}{\|s_k\|_2}.$$

By (2.1.2) and $\lambda_k \sigma_k < 0$ for every k, for any $j > 0$,

$$\begin{aligned} f(x_j) - f(x_0) &= \sum_{k=0}^{j-1} f(x_{k+1}) - f(x_k) \\ &\leq \sum_{k=0}^{j-1} \alpha \nabla f(x_k)^T s_k \\ &= \alpha \sum_{k=0}^{j-1} \lambda_k \sigma_k < 0. \end{aligned}$$

Hence, either

$$\lim_{j \to \infty} f(x_j) = -\infty \quad \text{or} \quad -\sum_{k=0}^{\infty} \lambda_k \sigma_k \text{ is convergent , i.e. } -\sum_{k=0}^{\infty} \lambda_k \sigma_k < \infty .$$

In the first case (b) is true and we are finished, so we consider the second. In particular, we deduce that $\lim_{k \to \infty} \lambda_k \sigma_k = 0$. Now we want to conclude that $\lim_{k \to \infty} \sigma_k = 0$, and so we need to use condition (2.1.3), since it was imposed to ensure that the steps don't get too small.

We have for each k that

$$\nabla f(x_{k+1})^T s_k \geq \beta \nabla f(x_k)^T s_k$$

and so

$$\left[\nabla f(x_{k+1}) - \nabla f(x_k)\right]^T s_k \geq (\beta - 1) \nabla f(x_k)^T s_k > 0,$$

by (2.2.4) and $\beta < 1$. Applying the Cauchy-Schwarz inequality and (2.2.3) to the left side of the last equation , and using the definition of σ_k , gives us

$$0 < (\beta - 1) \lambda_k \sigma_k \leq \gamma \left\| s_k \right\|^2 = \gamma \lambda_k^2 ,$$

so

$$\lambda_k \geq \frac{\beta - 1}{\gamma} \sigma_k > 0$$

and

$$\lambda_k \sigma_k \leq \frac{\beta - 1}{\gamma} \sigma_k^2 < 0 .$$

Thus

$$0 = \lim_{k\to\infty} \lambda_k \sigma_k \le \frac{\beta - 1}{\gamma} \lim_{k\to\infty} \sigma_k^2 \le 0,$$

which shows that

$$\lim_{k\to\infty} \sigma_k = 0,$$

i.e. (c) is true and completes the proof.//

The above theorem is also due to Wolfe .It shows that if any sequence $\{x_k\}$is generated to obey $\nabla f(x_k)^T (x_{k+1} - x_k) < 0$, (2.1.2) and (2.1.3) then, unless the angle between $\nabla f(x_k)$ and $(x_{k+1} - x_k)$ converges to $\frac{\pi}{2}$ as $k \to \infty$, either the gradient converges to 0, or f is unbounded below of course ,both may be true. We comment below that the case where the angle between $\nabla f(x_k)$ and $(x_{k+1} - x_k)$ approaches $\frac{\pi}{2}$ can be prevented.

Note that while Theorem 2.2.2 applies readily to any line-search algorithm, it is completely independent of the method for selecting the descent directions or the step lengths. Therefore, this theorem gives sufficient conditions for global convergence, in a weak sense, of any optimization algorithm, including the model-trust region algorithms on chapter 3.

Theorem 2.2.3 due to Dennis and Mor`e, shows that our global strategy will permit full Quasi-Newton steps $x_{k+1} := x_k - H_k^{-1} \nabla f(x_k)$ close to a minimizer of f as long as $-H_k^{-1} \nabla f(x_k)$ is close enough to the Newton step.

Theorem 2.2.3: Let $f:\mathrm{R}^n \to \mathrm{R}$ be twice continuously differentiable in an open convex set D, and assume that $\nabla^2 f \in Lip_\gamma(D)$ .Consider a sequence $\{x_k\}$ generated by $x_{k+1} := x_k + \lambda_k p_k$ where $\nabla f(x_k)^T p_k < 0$ for all k and λ_k is chosen to satisfy (2.1.2) with an $\alpha < \frac{1}{2}$, and (2.1.3). If $\{x_k\}$ converges to a point $x_* \in D$ at which $\nabla^2 f(x_*)$ is positive definite, and if

$$\lim_{k\to\infty} \frac{\left\|\nabla f(x_k) + \nabla^2 f(x_k) p_k\right\|_2}{\|p_k\|_2} = 0, \tag{2.2.5}$$

then there is an index $k_0 \geq 0$ such that for all $k \geq k_0$, $\lambda_k = 1$ is admissible.

Proof. The proof is really just a generalization of the easy exercise that if $f(x)$ is a positive definite quadratic and $p_k = -\nabla^2 f(x_k)^{-1} \nabla f(x_k)$, then $\nabla f(x_k)^T p_k < 0$ and $x_{k+1} = x_k + p_k$ satisfies (2.1.2) for any $\alpha \leq \frac{1}{2}$, and (2.1.3) for any $\beta \geq 0$.

We set

$$\rho_k = \frac{\left\|\nabla f(x_k) + \nabla^2 f(x_k) p_k\right\|}{\|p_k\|}. \tag{2.2.6}$$

First we show that

$$\lim_{k\to\infty} \|p_k\| = 0 .$$

By now familiar argument, if $\bar{x}$ is near enough x_*, then $\nabla^2 f(\bar{x})^{-1}$ exists and is positive definite, and if $\mu^{-1} = \left\|\nabla^2 f(x_*)^{-1}\right\|$, then $\frac{1}{2}\mu^{-1} \leq \left\|\nabla^2 f(\bar{x})^{-1}\right\| \leq 2\mu^{-1}$.

Therefore, since

$$-\frac{\nabla f(x_k)^T p_k}{\|p_k\|} = \frac{p_k{}^T \nabla^2 f(x_k) p_k}{\|p_k\|} - \frac{(\nabla f(x_k) + \nabla^2 f(x_k) p_k)^T p_k}{\|p_k\|}$$

and $p_k{}^T \nabla^2 f(x_k) p_k > \frac{1}{2}\mu \|p_k\|^2$, (2.2.6) and (2.2.5) show that for k sufficiently large,

$$-\frac{\nabla f(x_k)^T p_k}{\|p_k\|} \geq (\tfrac{1}{2}\mu - \rho_k)\|p_k\| \geq \tfrac{1}{4}\mu \|p_k\| . \tag{2.2.7}$$

Since from Theorem 2.2.2 we have

$$\lim_{k\to\infty}\frac{\nabla f(x_k)^T p_k}{\|p_k\|}=0,$$

The inequality (2.2.7) implies

$$\lim_{k\to\infty}\|p_k\|=0.$$

We now show that $\lambda_k=1$ satisfies (2.1.2), for all $k\geq k_0$. From the mean value theorem, for some $\bar{x}_k\in[x_k,x_k+p_k]$. We get

$$f(x_k+p_k)-f(x_k)=\nabla f(x_k)^T p_k+\tfrac{1}{2}p_k{}^T\nabla^2 f(\bar{x}_k)p_k$$

or

$$\begin{aligned}f(x_k+p_k)-f(x_k)-\tfrac{1}{2}\nabla f(x_k)^T p_k&\\ =\tfrac{1}{2}(\nabla f(x_k)+\nabla^2 f(\bar{x}_k)p_k)^T p_k&\\ =\tfrac{1}{2}(\nabla f(x_k)+\nabla^2 f(x_k)p_k)^T p_k+\tfrac{1}{2}p_k{}^T\left[\nabla^2 f(\bar{x}_k)-\nabla^2 f(x_k)\right]p_k&\\ \leq\tfrac{1}{2}(\rho_k+\gamma\|p_k\|)\|p_k\|^2 \qquad (2.2.8)&\end{aligned}$$

because of (2.2.6) and by the Lipschitz continuity of $\nabla^2 f$. Now choose k_0 so that for $k\geq k_0$, (2.2.7) holds and

$$\rho_k+\gamma\|p_k\|\leq\tfrac{1}{4}\mu\min(\beta,1-2\alpha). \qquad (2.2.9)$$

If $k\geq k_0$, then $\lambda_k=1$ satisfies (2.1.2) because from (2.2.8), (2.2.7), and (2.2.9) follows

$$\begin{aligned}f(x_k+p_k)-f(x_k)&\leq\tfrac{1}{2}\nabla f(x_k)^T p_k+\tfrac{1}{8}\mu(1-2\alpha)\|p_k\|^2\\ &\leq\tfrac{1}{2}(1-(1-2\alpha))\nabla f(x_k)^T p_k\\ &=\alpha\nabla f(x_k)^T p_k.\end{aligned}$$

To show that (2.1.3) is satisfied by $\lambda_k=1$ for $k\geq k_0$, we use the mean value theorem again to get, for some $z_k\in(x_k,x_k+p_k)$, that

$$\begin{aligned}\nabla f(x_k+p_k)^T p_k&=(\nabla f(x_k)+\nabla^2 f(z_k)p_k)^T p_k\\ &=(\nabla f(x_k)+\nabla^2 f(x_k)p_k)^T p_k+p_k{}^T(\nabla^2 f(z_k)-\nabla^2 f(x_k))p_k.\end{aligned}$$

This implies

$$\left|\nabla f(x_k + p_k)^T p_k\right| \le \rho_k \|p_k\|^2 + \gamma \|p_k\|^3$$
$$\le \frac{\mu\beta}{4}\|p_k\|^2$$
$$\le -\beta \nabla f(x_k)^T p_k$$

by (2.2.6), the Lipschitz continuity of $\nabla^2 f$, (2.2.9) and (2.2.7) . This yields $\nabla f(x_k + p_k)^T p_k \ge \beta \nabla f(x_k)^T p_k$ for $k \ge k_0$ .Thus $\lambda_k = 1$ eventually satisfies (2.1.2) and (2.1.3), and so it is admissible. Finally, The $\lim_{k\to\infty} p_k = 0$ and (2.2.5) implies that $\nabla f(x_*) = 0$ .//

2.3 STEP SELECTION BY BACKTRACKING

We now specify how our line search algorithm will choose λ_k .As we have stated, the modern strategy is to start with λ_k=1, and then, if $x_k + p_k$ is not acceptable, "backtrack"(reduce λ_k) until an acceptable $x_k + \lambda_k p_k$ is found. The framework of such an algorithm is given below.

ALGORITHM 2.3.1: BACKTRACKING LINE-SEARCH FRAMEWORK

Given $\alpha \in (0, \frac{1}{2})$, $0 < l < u < 1$ $\lambda_k = 1$.

While $f(x_k + \lambda_k p_k) > f(x_k) + \alpha \lambda_k \nabla f(x_k)^T p_k$, do

$\lambda_k := \rho\lambda_k$ for some $\rho \in [l,u]$;(* ρ is chosen a new each time by the line search*)

$x_{k+1} := x_k + \lambda_k p_k$

In practice, α is set quite mall, so that hardly more than a decrease in function value is required. Our algorithm uses $\alpha = 10^{-4}$.

2.3.1 STRATEGY FOR REDUCING λ_k (CHOOSING ρ)

Define $\hat{f}(\lambda) := f(x_k + \lambda p_k)$ which is the one-dimensional restriction of f to the line through x_k in the direction p_k. If we need to backtrack, we will use our most current information about $\hat{f}$ to model it, and then take the value of λ that minimizes this model as our next value of λ_k in Algorithm 2.3.1.

Initially, we have

$$\hat{f}(0) = f(x_k) \text{ and } \hat{f}'(0) = \nabla f(x_k)^T p_k \qquad (2.3.1)$$

$$\hat{f}(1) = f(x_k + p_k). \qquad (2.3.2)$$

So if $f(x_k + p_k)$ does not satisfy (2.1.2), that is $\hat{f}(1) > \hat{f}(0) + \alpha f'(0)$, then we model $\hat{f}(\lambda)$ by the one-dimensional quadratic model satisfying (2.3.1) and (2.3.2),

$$\hat{m}_q(\lambda) = [\hat{f}(1) - \hat{f}(0) - \hat{f}'(0)]\lambda^2 + \hat{f}'(0)\lambda + \hat{f}(0),$$

and calculate the point

$$\hat{\lambda} = \frac{-\hat{f}'(0)}{2[\hat{f}(1) - \hat{f}(0) - \hat{f}'(0)]} \qquad (2.3.3)$$

for which $\hat{m}_q'(\hat{\lambda}) = 0$.

Now

$$m_q''(\lambda) = 2[\hat{f}(1) - \hat{f}(0) - \hat{f}'(0)] > 0,$$

since $\hat{f}(1) > \hat{f}(0) + \alpha \hat{f}'(0) > \hat{f}(0) + \hat{f}'(0)$. Thus $\hat{\lambda}$ minimizes $\hat{m}_q(\lambda)$. Furthermore; $\hat{\lambda} > 0$, because $f'(0) < 0$. Therefore we take $\hat{\lambda}$ as our new value of λ_k in Algorithm 2.3.1.

Note that since $\hat{f}(1) > \hat{f}(0) + \alpha \hat{f}'(0)$, we have

$$\hat{\lambda} < \frac{1}{2(1-\alpha)}.$$

In fact, if $\hat{f}(1) \geq \hat{f}(0)$, then $\hat{\lambda} \leq \frac{1}{2}$. Thus (2.3.3) gives a useful implicit upper bound $u \approx \frac{1}{2}$ on the first value of ρ in Algorithm 2.3.1.On the other hand, if $\hat{f}(1)$ is much larger than $\hat{f}(0)$, $\hat{\lambda}$ can be very small but we impose a lower bound of $l = \frac{1}{10}$ in Algorithm 2.3.1. This means that at the first backtrack at each iteration, if $\hat{\lambda} \leq 0.1$, then we next try $\lambda_k = \frac{1}{10}$.

Example 2.3.2:(cf.[3]) Let $f:\mathrm{R}^n \to \mathrm{R}$ and

$$f(x_1, x_2) := x_1^4 + x_1^2 + x_2^2, \quad x_c = (1,1)^T, p_c = (-3,-1)^T.$$

Let $\alpha = 0.1$ in (2.1.2) and $\beta = 0.5$ in (2.1.3). Since $\nabla f(x_c)^T p_c = (6,2)(-3,-1)^T = -20 < 0$, p_c is a descent direction for $f(x)$ from x_c.

Now consider $x(\lambda) = x_c + \lambda p_c$.If $\lambda = 1$, we get $x_+ := x_c + p_c = (-2,0)^T$ and $\nabla f(x_+) p_c = (-36,0)(-3,-1)^T = 108 > -10 = \beta \nabla f(x_c)^T p_c$. This implies that x_+ satisfies (2.1.3), but $f(x_+) = 20 > 1 = f(x_c) + \alpha \nabla f(x_c)^T p_c$ so that x_+ does not satisfy (2.1.2).

Therefore x_+ is not acceptable and a backtrack is needed. Then

$$\hat{m}_q(0) = f(x_c) = 3, \hat{m}'_q(0) = \nabla f(x_c)^T p_c = -20, \hat{m}_q(1) = f(x_c + p_c) = 20,$$

and (2.3.3) gives $\hat{\lambda} = \frac{10}{37} \cong 0.270$.

Now $x_c + \hat{\lambda} p_c \cong (0.189, 0.730)^T$ satisfies condition

(2.1.2) with $\alpha = 10^{-4}$, since $f(x_c + \hat{\lambda} p_c) \cong 0.570 \leq 2.99946 = f(x_c) + \alpha \lambda \nabla f(x_c)^T p_c$.

Hence the next is $x_+ := x_c + \hat{\lambda} p_c$.Continuing the process, the minimum of $f(x_c + \lambda p_c)$ occurs at $\lambda_* \cong 0.40$.

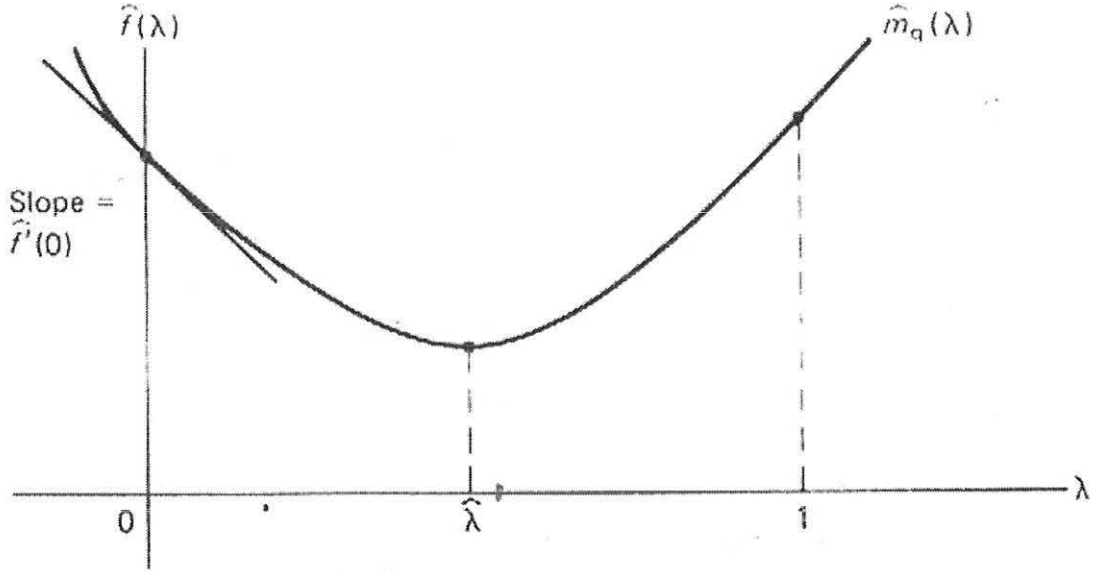

Figure 2.3.3

In Figure 2.3.3 we can see backtracking at the first iteration using a quadratic model.

Now suppose $\hat{f}(\lambda_k) = f(x_k + \lambda_k p_k)$ does not satisfy (2.1.2). In this case we need to backtrack again. Although we could use a quadratic model as we did on the first backtrack, we now have four informations about $\hat{f}$ which are $\hat{f}(0), \hat{f}'(0)$, and the last values of $\hat{f}(\lambda)$. So at this and any subsequent backtrack during the current iteration, we use a cubic model of $\hat{f}$. The calculation of λ proceeds as follows.

Let λprev and λ2prev be the last two previous values of λ_k.Then the cubic that satisfies $\hat{f}(0), \hat{f}'(0), \hat{f}(\lambda prev)$, and $\hat{f}(\lambda 2 prev)$ is

$$\hat{m}_{cu}(\lambda) = a\lambda^3 + b\lambda^2 + \hat{f}'(0)\lambda + \hat{f}(0),$$

where

$$\begin{pmatrix} a \\ b \end{pmatrix} = \frac{1}{\lambda prev - \lambda 2 prev} \begin{pmatrix} \frac{1}{\lambda prev^2} & \frac{-1}{\lambda 2 prev^2} \\ \frac{-\lambda 2 prev}{\lambda prev^2} & \frac{\lambda prev}{\lambda 2 prev^2} \end{pmatrix} \begin{pmatrix} \hat{f}(\lambda prev) - \hat{f}(0) - \hat{f}'(0)\lambda prev \\ \hat{f}(\lambda 2 prev) - \hat{f}(0) - \hat{f}'(0)\lambda 2 prev \end{pmatrix}$$

Its local minimizing point is $\hat{\lambda} = \dfrac{-b + \sqrt{b^2 - 3a\hat{f}'(0)}}{3a}$. (2.3.4)

It can be shown that $\hat{\lambda}$ is always real, if $\alpha < \frac{1}{4}$ in Algorithm (2.3.1). Finally, we use an upper bound $u = 0.5$ and a lower bound $l = \frac{1}{10}$ for $\hat{\lambda}$. This means that if $\hat{\lambda} > \frac{1}{2}\lambda prev$, we set the new $\lambda_k = \frac{1}{2}\lambda prev$ and if $\hat{\lambda} < \frac{1}{10}\lambda prev$, we set the new $\lambda_k = \frac{1}{10}\lambda prev$.

3.MODEL -TRUST REGION

3.1 THE MODEL -TRUST REGION APPROACH

The underlying assumptions in line-search approach were that the direction would be the Quasi-Newton direction, and that the full Quasi-Newton step would always be the first trail step. The resulting backtracking algorithm incorporated these assumptions to attempt global convergence without sacrificing the local convergence properties of the Newton method, clearly the goal of any global strategy. In this approach we see the same goal, but we drop the assumption that shortened steps must be in the Quasi-Newton direction.

Suppose that the full Quasi-Newton step is unsatisfactory. This indicates that our quadratic model does not adequately model f in a region containing the full Quasi-Newton step. In line –search algorithms we retain the same step direction and choose a shorter *step length*. In this trust region approach, when we need to take a shorter step, we first will choose a shorter *step length*, and then choose the *step direction*.

3.1.1 SELECTION OF A STEP OF MAXIMAL LENGTH δ_C FROM x_C.

Now suppose that we have x_c and some estimate δ_c of the maximum length of a successful step we are likely to be able to take from x_c. This raises the question: How can we best select a step of maximal length δ_c from x_c? From the beginning we have taken the view that the Quasi-Newton step s_c^N is reasonable because it is the step from x_c to the global minimizer of the local quadratic model m_c (if the model Hessian H_c is positive definite). If we add the idea of bounding

the maximal step length by $\delta_c > 0$, then the corresponding answer to our question is to try the step s_c that solves

$$\begin{aligned} &\min m_c(x_c + s) = f(x_c) + \nabla f(x_c)^T s + \tfrac{1}{2} s^T H_c s \\ &\|s\|_2 \le \delta_c. \end{aligned} \tag{3.1.1}$$

Problem (3.1.1) is the basis of the "model -trust region" approach to minimization.

Its solution is given in Lemma 3.1.1 below. The name comes from viewing δ_c as providing a region in which we can trust m_c to adequately model f.

Lemma 3.1.1: Let

1. $f : \mathrm{R}^n \to \mathrm{R}$ be twice continuously differentiable.
2. $H_c \in \mathrm{R}^{n \times n}$ be symmetric and positive definite.

Then

a. If $\|s(0)\| \le \delta_c$, then $s(0) = s_c^N$ is the solution of problem (3.1.1)

b. If $\|s(0)\| > \delta_c$, then problem (3.1.1) is solved by

$s(\mu) := -(H_c + \mu I)^{-1} \nabla f(x_c)$ FOR the unique $\mu \ge 0$ such that $\|s(\mu)\| = \delta_c$

(3.1.2)

c. for any $\mu \ge 0$, then $s(\mu)$ defines a descent direction for f from x_c.

Proof. Let us call the solution to (3.1.1) s_*, and let $x_* = x_c + s_*$. Since the Newton point $x_c + s_c^N$ is the global minimizer of m_c, it is clear that if $\|s_c^N\| \le \delta_c$, then $s_c^N = s_*$. Now consider the case when $x_c + s_c^N$ is outside the step bound as in figure 3.1.2.

Let $\bar{x}$ be any point interior to the constraint region. That is $\|\bar{x} - x_c\| < \delta_c$. Then $\nabla m_c(\bar{x}) \ne 0$, so it is possible to decrease m_c from $\bar{x}$ while staying inside the constrained region by considering points of the from $\bar{x} - \lambda \nabla m_c(\bar{x})$.

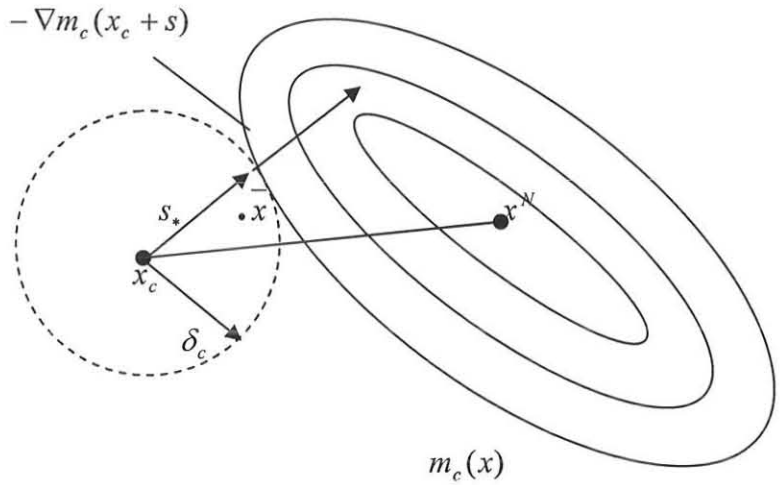

Figure 3.1.2

In Figure 3.1.2 we see x_c, m_c with minimum at x^N surrounded by contours at some increasing values of m_c, and a step bound δ_c.

This implies that $\bar{s}$ can not be the solution to (3.1.1), so s_* must satisfy $\|s_*\| = \delta_c$ unless $\|s(0)\| < \delta_c$. Also since the constrained region in problem (3.1.1) is closed and compact, some s for which $\|s\| = \delta_c$ must be the solution.

Now consider some s such that $\|s\| = \delta_c$. For s to be the solution to (3.1.1), it is necessary that if we move an arbitrary small distance from $x_c + s$ in any descent direction for m_c, we increase the distance from x_c. A descent direction for m_c from $x_c + s$ is any vector p for which

$$0 > p^T \nabla m_c(x_c + s) = p^T [H_c s + \nabla f(x_c)] \ . \qquad (3.1.3)$$

Similarly, a direction $\bar{p}$ from $x_c + s$ increase distance from x_c if and only if

$$\bar{p}^T s > 0 \, . \qquad (3.1.4)$$

What we are saying is that for s to solve (3.1.1), any p that satisfies (3.1.3) must also satisfy (3.1.4). Since $\nabla m_c(x_c + s) \neq 0$, it implies that for some $\mu > 0$,

$\mu s = -\nabla m_c(x_c + s) = -(H_c s + \nabla f(x_c))$ Which is just $(H_c + \mu I)s = -\nabla f(x_c)$.Thus $s_* = s(\mu) = -(H_c + \mu I)^{-1}\nabla f(x_c)$ for some $\mu > 0$, when $\|s_c^N\| > \delta_c$. Since $\mu \geq 0$ and H_c is symmetric and positive definite, $(H_c + \mu I)$ is symmetric and positive definite. Therefore,

$s(\mu) = -(H_c + \mu I)^{-1}\nabla f(x_c)$ is a descent direction for f from x_c. In order to finish the proof, we need to show that s_* is unique. If $\mu \geq 0$ and $\eta(\mu) := \|s(\mu)\|_2 = \|(H_c + \mu I)^{-1}\nabla f(x_c)\|$,

then

$$\eta'(\mu) = \frac{-\nabla f(x_c)^T (\mu I + H_c)^{-3}\nabla f(x_c)}{\|s(\mu)\|} = \frac{-s(\mu)^T (H_c + \mu I)^{-1} s(\mu)}{\|s(\mu)\|}$$

and so $\eta'(\mu) < 0$ when $\nabla f(x_c) \neq 0$. Thus, if $\mu_1 > \mu_2 \geq 0$, then $\|s(\mu_1)\| < \|s(\mu_2)\|$ .This shows that s_* is unique.

Lemma 3.1.1 is a fundamental for a step a minimization algorithm. But the difficulty is that there is no finite method of determining the $\mu_c > 0$ with $\|s(\mu_c)\|_2 = \delta_c$ in the case of $\delta_c < \|H_c^{-1}\nabla f(x_c)\|_2 = \|s(0)\|_2$.

3.2 MODEL TRUST REGION ALGORITHM

There is no guarantee that the x_+ will be an acceptable next point, although we hope it will be if δ_c is a good step bound. If x_+ is unacceptable, one reduces the size of the trust region and minimizes the same quadratic model on the smaller trust region. Therefore, a complete step of a trust –region algorithm will have the following form:

Algorithm 3.2.1: A global step by the model –trust region approach

Given $f:\mathbb{R}^n \to \mathbb{R}$, $\delta_c > 0$, $x_c \in R^n$, $H_c \in R^{n\times n}$ a symmetric and positive definite:

(1) $s_c :=$ approximate solution to (3.1.1),

$x_+ := x_c + s_c$

(2) If x_+ is unacceptable, then calculate a new value δ_c and repeat (1).

(3) If x_+ is acceptable, then use

$\delta_+ := \delta_c$

3.3 COMPUTATIONAL METHODS TO DETERMINE μ_c

3.3.1 THE LOCALLY CONSTRAINED OPTIMAL ("HOOK") STEP

Our first approach for calculating the step, when $\|s(0)\| > \delta_c$, is to find $s(\mu) := -(H_c + \mu I)^{-1}\nabla f(x_c)$ such that $\|s(\mu)\| \approx \delta_c$, and then make a trial step to $x_+ := x_c + s(\mu)$. $\Phi:\ \mathbb{R}\to\mathbb{R}$ be defined by

$$\Phi(\mu) = \|s(\mu)\| - \delta_c = 0 \qquad (3.3.1)$$

In this case, for n=1 we take $H_c =: h_c \in \mathbb{R}, \nabla f(x_c) = g_c$ which gives $\Phi(\mu) = \left| -(\mu + h_c)^{-1} g_c \right| - \delta_c$. This suggests a local model of the form

$$m_c(\mu) = \frac{\alpha_c}{\beta_c + \mu} - \delta_c .$$

Notice that, we will use boldface for the current values of the quantities α_c, β_c, μ_c that are changing in the inner iteration on μ, and normal type for the current values of $\delta_c, x_c, \nabla f(x_c), H_c$ that come from the outer iteration and are unchanged during the

solution of (3.3.1). Thus, the inner iteration's last approximation μ_c to μ_* is the exact solution to (3.3.1).

The model $m_c(\mu)$ has two free parameters α_c, β_c. Therefore it is reasonable to choose them to satisfy the two conditions:

$$m_c(\mu) = \frac{\alpha_c}{\beta_c + \mu_c} - \delta_c = \Phi(\mu) = \|s(\mu_c)\| - \delta_c \text{ and}$$

$$m_c'(\mu_c) = -\frac{\alpha_c}{(\beta_c + \mu_c)^2} = \Phi'(\mu_c) = -\frac{s(\mu_c)^T (H_c + \mu_c I)^{-1} s(\mu_c)}{\|s(\mu_c)\|_2}$$

This gives

$$\alpha_c = -\frac{(\Phi(\mu_c) + \delta_c)^2}{\Phi'(\mu_c)} \qquad (3.3.2)$$

$$\beta_c = -\frac{(\Phi(\mu_c) + \delta_c)}{\Phi'(\mu_c)} - \mu_c \qquad (3.3.3)$$

Now that we have our model $m_c = (\mu)$, we naturally choose μ_+ such that $m_c(\mu) = 0$

That is $\mu_+ = \frac{\alpha_c}{\delta_c} - \beta_c$

Substituting (3.3.3), (3.3.2) and (3.3.1) in to this,

we get

$$\mu_+ = \mu_c - \left[\frac{\Phi(\mu_c) + \delta_c}{\delta_c}\right]\left[\frac{\Phi(\mu_c)}{\Phi'(\mu_c)}\right] \qquad (3.3.4)$$

$$= \mu_c - \frac{\|s(\mu_c)\|}{\delta_c}\left[\frac{\Phi(\mu_c)}{\Phi'(\mu_c)}\right]$$

as our iteration for solving $\Phi(\mu) = 0$, and remember where we are in the iteration: We have just made an x- step $s(\mu_-)$ from x_- to x_c, and we now want x_+, i.e. we have obtained δ_c from δ_- by updating the trust region. Before we get H_c, we still have $(H_- + \mu_- I)$ in factored form and so it costs only $O(n^2)$ to compute

$\Phi := \|s(\mu_-)\| - \delta_c$ and $\Phi' := \frac{s(\mu_-)^T (H_- + \mu_- I)^{-1} s(\mu_-)}{\|s(\mu_-)\|}$.

In analogy with (3.3.4), we use

$$\mu_0 = \mu_- - \frac{\|s(\mu_-)\|}{\delta_c}\frac{\Phi(\mu_-)}{\Phi'(\mu_-)}. \qquad (3.3.5)$$

Notice that if the previous iteration took the Newton step ($\mu_- = 0$), then we find μ_0 by a different technique mentioned latter.

Another computational detail, important to the performance of the algorithm for solving (3.3.1) is the generation and updating of lower and upper bounds u_+ and l_+ on μ_+ which used to safeguard the steps. i.e. we restrict μ_+ to be in $[l_+,u_+]$.We take $l_0 = \frac{-\Phi(0)}{\Phi'(0)}$, we then calculate

$$\mu_+^N = \mu_c - \frac{\Phi(\mu_c)}{\Phi'(\mu_c)}$$

along with each calculation of (3.3.4), and update the lower bound to $l_+=\max\{l_c, \mu_+^N\}$,where l_c is the current lower bound. Also, we take $u_0=\frac{\|\nabla f(x_c)\|_2}{\delta_c}$ as our initial upper bound. Then, at each iteration, if $\Phi(\mu_c)<0$, update the upper bound to $u_+= \min\{u_c, \mu_c\}$, where u_c is the current upper bound. If at any iteration

$\mu_+ \notin [l_+,u_+]$ we take μ_+ by $\mu_+=\max\{(l_+.u_+)^{1/2}, 10^{-3}u_+\}$ (3.3.6)

Notice (3.3.6) is used to define μ_0 whenever (3.3.5) can not be used because the previous iteration used the Newton step.

Finally, we don't solve (3.3.5) to any great accuracy; However, if $\|s(\mu)\|_2 \in \left[\frac{3\delta_c}{4}, \frac{3\delta_c}{2}\right]$, we will accept the result as solution for (3.3.5). This is because the trust region is never increased or decreased by a factor smaller than 2.

Example 3.3.1: (cf.[3]) Let $f(x_1,x_2)=x_1^4+x_1^2+x_2^2$, $x_c=(1,1)^T$, $H_c=\nabla^2 f(x_c)$,

$\delta_c=\frac{1}{2}$, $\mu_-=0$, then

$$\nabla f(x_c)=(6,2)^T \quad H_c=\begin{pmatrix}14 & 0\\ 0 & 2\end{pmatrix}$$

$$s_c^N=-H_c^{-1}\nabla f(x_c)=(-\frac{3}{7},-1)^T,\ \left\|s_c^N\right\|\approx 1.088.$$

Since $\left\|s_c^N\right\| > \frac{3}{2}\delta_c$, the Newton step is too long, and we seek some $\mu>0$

Such that

$$\left\|(H_c+\mu I)^{-1}\nabla f(x_c)\right\| \in\left[\frac{3\delta_c}{4},\frac{3\delta_c}{2}\right]=[0.375,0.75] \text{ (see figure 3.3.2).}$$

Since $\mu_-=0$ we get

$$l_0=0-\frac{\Phi(0)}{\Phi'(0)}\approx\frac{0.588}{0.472}\approx 1.25$$

$$u_0=\frac{\left\|\nabla f(x_c)\right\|_2}{\delta_c}\approx\frac{6.325}{0.5}\approx 12.6$$

$$\mu_0=(l_0.u_0)^{\frac{1}{2}}\approx 3.97.$$

Next we calculate

$$s(\mu_0)=-(H_c+\mu_0 I)^{-1}\nabla f(x_c)$$
$$\approx(-0.334,-0.335)^T.$$

Since $\left\|s(\mu_0)\right\|_2\approx 0.473\in[0.375,0.75]$,

we take $\mu_c=\mu_0$ and $x_+=x_c+s(\mu_c)\cong(0.666,0.665)^T$ as our approximate solution to (3.2.1). Note that in this case we have not used iteration (3.3.4). One application of (3.3.4) would result in

$$\mu_1=\mu_0-\frac{\Phi(\mu_0)}{\Phi'(\mu_0)}\frac{\left\|s(\mu_0)\right\|_2}{\delta_c}=3.97-\frac{-0.0271}{-0.0528}\frac{+0.473}{+0.5}=3.49.$$

and that $s(\mu_1) = (-0.343,-0.365)^T$, $\|s(\mu_1)\|_2 = 0.5006$. Incidentally, (3.3.1) is solved exactly by $\mu_* \cong 3.496$.

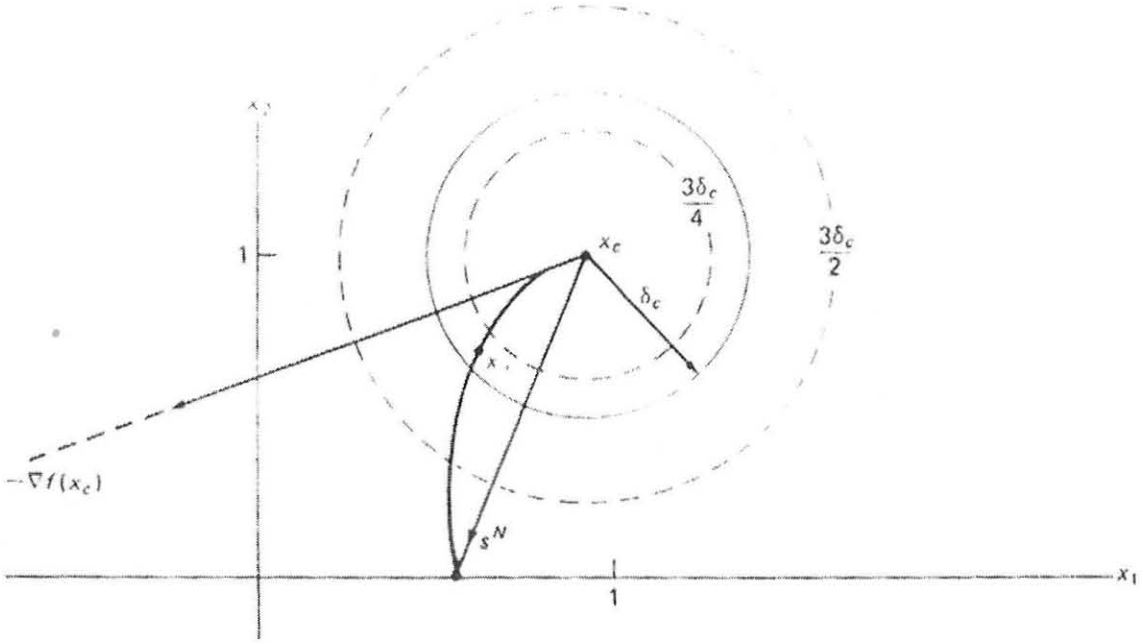

Figure 3.3.2

The figure 3.3.2 shows the "Hook" step for Example 3.3.1.

3.3.2 THE DOUBLE DOGLEG STEP

The double dogleg step method also finds the solution to problem (3.3.3). It approximate the solution by a piecewise linear function connecting the "Cauchy point "C.P., the minimizer of the quadratic model m_c in the steepest-descent direction ,to the Newton direction for m_c, as indicated in the figure(3.3.3).Then x_+ chosen to be the point such that $\|x_+ - x_c\|_2 = \delta_c$, unless $\left\|H_c^{-1}\nabla f(x_c)\right\|_2 < \delta_c$, in which case x_+ is the Newton point.

The specific way of choosing the double dogleg curve have two important properties. First , as one proceeds along the piecewise linear curve from x_c to C.P.,

C.P. to $\hat{N}$ and $\hat{N}$ to x_+^N, the distance from x_c increases monotonically. Thus for any $\delta_c \le \left\| H_c^{-1} \nabla f(x_c) \right\|$, there is a unique point x_+ on the curve such that $\left\| x_+ - x_c \right\|_2 = \delta$. Second, the value of the quadratic model $m_c(x_c + s)$ decreases monotonically as s goes along the curve from x_c to C.P., C.P to $\hat{N}$ and $\hat{N}$ to x_+^N. This makes the process reasonable.

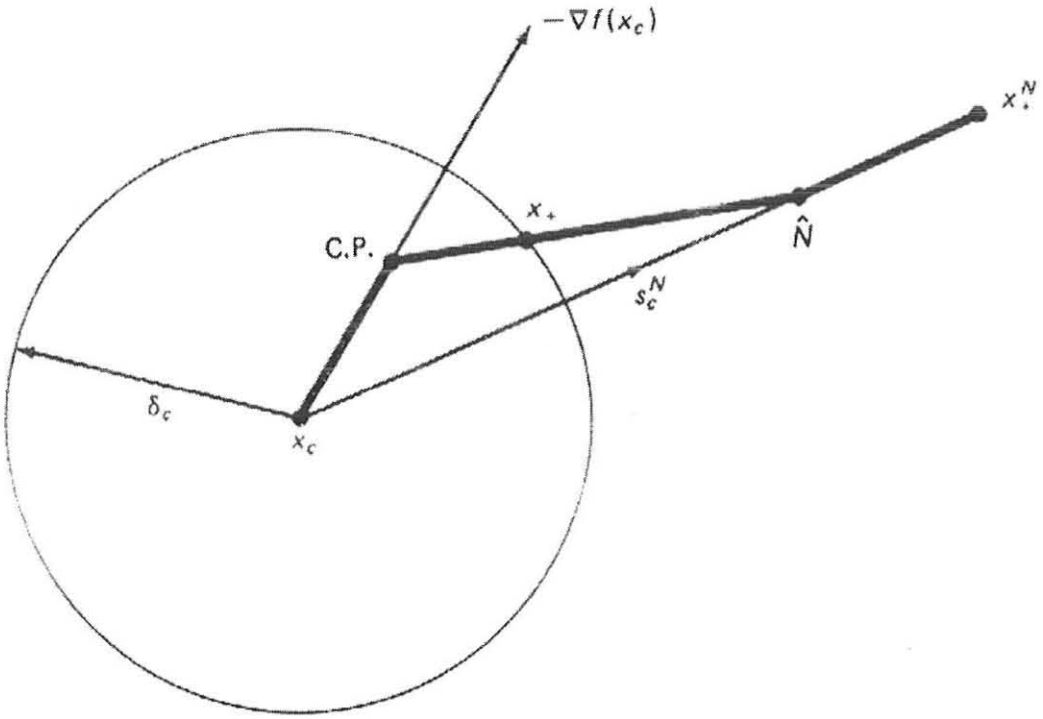

Figure 3.3.3

In Figure 3.3.3 we show the double dogleg curve, $x_c \to c.p. \to \hat{N} \to x_+^N$.

The Cauchy point C.P. is found by solving

$$\min_{\lambda \in \Re} m_c(x_c - \lambda \nabla f(x_c)) = f(x_c) - \lambda \left\| \nabla f(x_c) \right\|_2^2 + \tfrac{1}{2} \lambda^2 \nabla f(x_c)^T H_c \nabla f(x_c),$$

which has the unique solution

$$\lambda_* = \frac{\|\nabla f(x_c)\|_2^2}{\nabla f(x_c)^T H_c \nabla f(x_c)}.$$

Therefore,

$$\text{C.P.} = x_c - \lambda_* \nabla f(x_c),$$

and if

$$\delta_c \le \lambda_* \|\nabla f(x_c)\|_2 = \frac{\|\nabla f(x_c)\|_2^3}{\nabla f(x_c)^T H_c \nabla f(x_c)},$$

the algorithm takes a step of length δ_c in the steepest –descent direction:

$$x_+ = x_c - \frac{\delta_c}{\|\nabla f(x_c)\|_2} \nabla f(x_c).$$

In order for double dogleg curve to satisfy the first property stated above, it must be shown that the Cauchy point C.P is no farther from x_c than the Newton point x_+^N.

Let

$$s^{c.p} := -\lambda_* \nabla f(x_c), \; s^N := -H_c^{-1} \nabla f(x_c).$$

Then

$$\begin{aligned} \|s^{c.p}\|_2 &= \frac{\|\nabla f(x_c)\|_2^2}{\nabla f(x_c) H_c \nabla f(x_c)} \\ &\le \frac{\|\nabla f(x_c)\|_2^3}{\nabla f(x_c) H_c \nabla f(x_c)} \frac{\|\nabla f(x_c)\|_2 \|H_c^{-1} \nabla f(x_c)\|}{\nabla f(x_c) H_c^{-1} \nabla f(x_c)} \\ &= \frac{\|\nabla f(x_c)\|_2^4}{(\nabla f(x_c) H_c \nabla f(x_c)) \nabla f(x_c) H_c^{-1} \nabla f(x_c)} \|s^N\|_2 := \gamma \|s^N\|_2. \end{aligned} \tag{3.3.7}$$

But

$\frac{\|\nabla f(x_c)\|_2^4}{(\nabla f(x_c) H_c \nabla f(x_c)) \nabla f(x_c) H_c^{-1} \nabla f(x_c)} \le 1$, for a symmetric and positive definite $H \in \mathbb{R}^{n \times n}$.

Thus,

$$\|s^{c.p}\| \le \gamma \|s^N\| \le \|s^N\|$$

The point $\hat{N}$ on the double dogleg curve is now chosen to have the form

$$\hat{N} = x_c - \eta H_c^{-1} \nabla f(x_c)$$

for some η such that $\gamma \le \eta \le 1$ and

$m_c(x)$ decreases monotonically along the line from C.P to $\hat{N}$. (3.3.8)

Since we know that $m_c(x)$ decreases monotonically from x_c to C.P. and from $\hat{N}$ to x_+^N, (3.3.8) will guarantee that $m_c(x)$ decreases monotonically along the entire double dogleg curve.

To satisfy (3.3.8), η must be chosen so that the directional derivative along the line connecting C.P and $\hat{N}$ is negative at every point on that line segment. Parameterize this line segment by

$$x_+(\lambda) = x_c + s^{c.p.} + \lambda(\eta s^N - s^{c.p.}) \quad \text{where } 0 \le \lambda \le 1.$$

The directional derivative of m_c along this line at $x_+(\lambda)$ is

$$\begin{aligned}\nabla m_c(x_+(\lambda))^T(\eta s^N - s^{c.p.}) &= [\nabla f(x_c) + H_c(s^{c.p.} + \lambda(\eta s^N - s^{c.p.}))]^T(\eta s^N - s^{c.p.}) \\ &= (\nabla f(x_c) + H_c s^{c.p.})^T(\eta s^N - s^{c.p.}) + \\ &\qquad \lambda(\eta s^N - s^{c.p.})^T H_c(\eta s^N - s^{c.p.}).\end{aligned} \tag{3.3.9}$$

Since H_c is positive definite, the right hand side of (3.3.9) is monotonically increasing function of λ. Therefore we need only require that (3.3.9) be negative for $\lambda = 1$ to make it negative for $0 \le \lambda \le 1$. But for $\lambda = 1$ we get

$$\begin{aligned}&(\nabla f(x_c) + H_c s^{c.p.})^T(\eta s^N - s^{c.p.}) + \lambda(\eta s^N - s^{c.p.})^T H_c(\eta s^N - s^{c.p.}) \\ &= \nabla f(x_c)^T(\eta s^N - s^{c.p.}) + (H_c s^{c.p.})^T(\eta s^N - s^{c.p.}) + \\ &\qquad (\eta s^N H_c)^T(\eta s^N - s^{c.p.}) - (H_c s^{c.p.})^T(\eta s^N - s^{c.p.}) \\ &= \nabla f(x_c)^T(\eta s^N - s^{c.p.}) + (\eta s^N H_c)^T(\eta s^N - s^{c.p.}) \\ &= \nabla f(x_c)^T(\eta s^N - s^{c.p.}) - \eta \nabla f(x_c)^T(\eta s^N - s^{c.p.}), \text{ for } s^N = -H_c^{-1}\nabla f(x_c) \\ &= (1-\eta)\nabla f(x_c)^T(\eta s^N - s^{c.p.})\end{aligned}$$

$$= (1-\eta)\nabla f(x_c)^T \left(-\eta H_c^{-1}\nabla f(x_c) + \frac{\|\nabla f(x_c)\|^2 \nabla f(x_c)}{\nabla f(x_c) H_c \nabla f(x_c)} \right),$$

$$\text{for } s^N = -H_c^{-1}\nabla f(x_c) \text{ and}$$

$$s^{c.p.} = -\lambda_* \nabla f(x_c) = \frac{-\|\nabla f(x_c)\|^2 \nabla f(x_c)}{\nabla f(x_c) H_c \nabla f(x_c)}$$

$$= (1-\eta)\nabla f(x_c)^T \left(-\eta H_c^{-1}\nabla f(x_c) + \frac{\|\nabla f(x_c)\|^4 \nabla f(x_c)^{-1}}{\nabla f(x_c) H_c \nabla f(x_c)} \right)$$

$$= (1-\eta)\nabla f(x_c)^T \left(-\eta H_c^{-1}\nabla f(x_c) + \frac{\|\nabla f(x_c)\|^4}{\nabla f(x_c) H_c \nabla f(x_c)} \frac{H_c^{-1}\nabla f(x_c)}{\nabla f(x_c) H_c^{-1}\nabla f(x_c)} \right)$$

$$= (1-\eta)(\gamma-\eta)\nabla f(x_c) H_c^{-1}\nabla f(x_c).$$

Therefore the condition

$$\nabla m_c(x_+(1))^T (\eta s^N - s^{c.p.}) < 0$$

is equivalent to

$$(1-\eta)(\gamma-\eta)\nabla f(x_c) H_c^{-1}\nabla f(x_c) < 0$$

which is satisfied for any $\eta \in (\gamma, 1)$.

Thus $\hat{N}$ can be chosen as any point in the Newton direction $x_c + \eta s^N$ for which η is between 1 and γ, where

$$\gamma := \frac{\|\nabla f(x_c)\|_2^4}{\nabla f(x_c) H_c \nabla f(x_c) \nabla f(x_c) H_c^{-1}\nabla f(x_c)}.$$

Notice that if $\eta = 1$, we get the single dogleg curve . But an earlier bias to the Newton direction seems to improve the performance of the algorithm, and hence Dennis and Mei suggest that $\eta = 0.8\gamma + 0.2$, leading the double dogleg curve as figure 3.3.3.

Example 3.3.4: (cf.[3])Let $f(x), x_c, H_c$ be given as in Example 3.3.1, and let $\delta_c = 0.75$.

Recall that

$$\nabla f(x_c) = (6,2)^T,\ H_c = \begin{bmatrix} 14 & 0 \\ 0 & 2 \end{bmatrix},\quad s_c^N = (-\tfrac{3}{7}, -1)^T.$$

Since $\|s_c^N\|_2 = 1.088 > \delta_c$, the double dogleg algorithm first calculates the step to the Cauchy point . Now

$$s^{c.p.} = -\left(\frac{40}{512}\right)\begin{pmatrix} 6 \\ 2 \end{pmatrix} \approx \begin{pmatrix} -0.469 \\ -0.156 \end{pmatrix}.$$

Since $\|s^{c.p}\|_2 \approx 0.494 < \delta_c$, the algorithm next calculates the step to the point $\hat{N}$.

It can be verified that

$$\gamma = \frac{(40)^2}{(512)(\frac{32}{7})} \approx 0.684$$

$$\eta = 0.8\gamma + 0.2 \approx 0.747$$

$$s^{\hat{N}} := \eta s_c^N \approx \begin{pmatrix} -0.320 \\ -0.747 \end{pmatrix}$$

Since $\left\|s^{\hat{N}}\right\| \approx 0.813 > \delta_c$, the double dogleg step must be along the line connecting C.P. and $\hat{N}$ i.e. $s_c = s^{c.p.} + \lambda(s^{\hat{N}} - s^{c.p.})$ for the $\lambda \in (0,1)$, $\|s_c\| = \delta_c$. The number λ is calculated by solving for the positive root of the quadratic equation

$$\left\|s^{c.p.} + \lambda(s^{\hat{N}} - s^{c.p.})\right\|_2^2 = \delta_c^2,$$

Which is $\lambda \cong 0.867$. Thus,

$$s_c = s^{c.p} + 0.867(s^{\hat{N}} - s^{c.p}) \approx \begin{pmatrix} -0.340 \\ -.669 \end{pmatrix}$$

and

$$x_+ = x_c + s_c \approx \begin{pmatrix} 0.660 \\ 0.331 \end{pmatrix}.$$

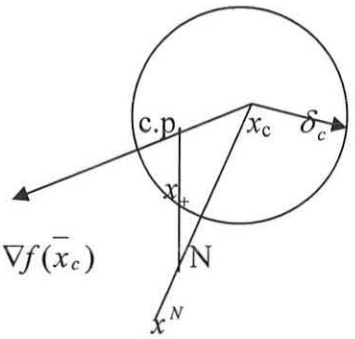

Figure 3.3.5

In Figure 3.3.5 the double dogleg step of Example 3.3.4 is shown.

3.4 UPDATING THE TRUST REGION

If the point x_+ found by using the techniques of subsection 3.3.1 or 3.3.2 is unacceptable, one reduces the size of the trust region and minimizes the same quadratic model on the smaller trust region. If x_+ is satisfactory, one must decide whether the trust region should be increased, decreased, or kept the same for the next step.

The condition for accepting x_+ is the one developed in Section 2.1,i.e.

$$f(x_+) \le f(x_c) + \alpha \nabla f(x_c)^T (x_+ - x_c) \qquad (3.4.1)$$

where α is a constant in $(0, \frac{1}{2})$ which is usually $\alpha = 10^{-4}$. If x_+ does not satisfy (3.4.1), we reduce the trust region by a factor between $\frac{1}{10}$ and $\frac{1}{2}$ and return to the approximate solution of the locally constrained minimization problem. The reduction factor is determined by the same quadratic backtrack strategy used to decrease the line-search parameter in Algorithm2.3.1. We model $f(x_c + \lambda(x_+ - x_c))$ by the quadratic $m_q(\lambda)$ that fits $f(x_c), f(x_+)$, and the directional derivative

$\nabla f(x_c)^T(x_+ - x_c)$ of f at x_c in the direction $x_+ - x_c$. We then let the new trust radius δ_+ extend to the minimizer of this model ,which occurs at

$$\lambda_* = \frac{-\nabla f(x_c)^T(x_+ - x_c)}{2(f(x_+) - f(x_c) - \nabla f(x_c)^T(x_+ - x_c))}$$

Thus, $\delta_+ = \lambda_* \|x_+ - x_c\|$. If

$$\lambda_* \|x_+ - x_c\|_2 \notin \left[\frac{1}{10}\delta_c, \frac{1}{2}\delta_c\right],$$

we instead set δ_+ to the closer endpoint of this interval for the same reasons as in the line-search algorithm.

Now suppose x_+ that satisfies (3.4.1). We need update δ_c to δ_+ by comparing the actual reduction $\Delta f := f(x_+) - f(x_c)$ to the predicted reduction $\Delta f_{pred} := m_c(x_+) - f(x_+)$ as follows:

$$\delta_+ = \begin{cases} 2\delta_c, & \Delta f \geq 0.75\Delta f_{pred} \\ \frac{\delta_c}{2}, & \Delta f < 0.1\Delta f_{pred} \\ \delta_c, & \text{otherwise.} \end{cases}$$

3.5 GLOBAL METHODS FOR SYSTEMS OF NON-LINEAR EQUATIONS

We consider the following problem to solve a system of non linear equations.

Let F: $\mathbb{R}^n \to \mathbb{R}^n$. Then find $x_* \in \mathbb{R}^n$ such that $F(x_*)=0$. (3.5.1)

In this section we show how Newton's method for (3.5.1) can be combined with global methods for unconstrained optimization to produce global methods for (3.1.1).

The Newton step for (3.5.1) is

$$x_+ = x_c - J(x_c)^{-1}F(x_c), \qquad (3.5.2)$$

where $J(x_c)$ is the Jacobian matrix of F at x_c.

From Section 1.1 we know that (3.5.3) is locally q-quadratically convergent to x_*, but not necessary globally convergent. Now assume x_c is not close to any solution x_* of (3.5.1). How would one decide then, whether to accept x_+ as the next iterate? A reasonable answer is that $\|F(x_+)\|$ should be less than $\|F(x_c)\|$ for some norm $\|.\|$, a convenient choice being the l_2 norm $\|F(x)\|^2 = F(x)^T F(x)$.

Let $f(x) := \frac{1}{2}\|F(x)\|$

Therefore, we have in effect turned our attention to the corresponding minimization problem:

$$\min_{x \in R^n} f(x). \qquad (3.5.3)$$

The descent direction for problem (3.5.3) is any direction p for which $\nabla f(x_c)^T p < 0$, where

$$\nabla f(x_c) = \frac{d}{dx}\sum_{i=1}^{n}\frac{1}{2}(f_i(x_c))^2 = \sum_{i=1}^{n}\nabla f_i(x_c).f(x_c) = J(x_c)^T F(x_c).$$

Therefore, the Newton direction along $s^N = -J(x_c)^{-1}F(x_c)$ is a descent direction, since

$$\nabla f(x_c)^T s^N = -F(x_c)^T J(x_c)J(x_c)^{-1}F(x_c) = -F(x_c)^T F(x_c) < 0$$

as long as $F(x_c) \neq 0$. The application of the global methods of Chapter 2 and Chapter 3 to the non linear equations is now straight forward. As long as $J(x_c)$ is sufficiently well conditioned, then $J(x_c)^T J(x_c)$ is safely positive definite, and the Algorithm 2.1.1 and Algorithm 3.2.1 apply without change if we define the objective function by $\frac{1}{2}\|F(x)\|^2$, the Newton direction by $-J(x_c)^{-1}F(x_c)$, and the positive definite quadratic model by,

$$\hat{m}_c(x_c + s) := \tfrac{1}{2}F(x_c)^T F(x_c) + (J(x_c)^T F(x_c))^T s + \tfrac{1}{2}s^T (J(x_c)^T J(x_c))s. \qquad (3.5.4)$$

This means that in a line-search algorithm, we search in the Newton direction, looking for a sufficient decrease in $\|F(x)\|$. In trust region algorithms, we consider the optimization problem :

$$\min \hat{m}_c(x_c + s)$$

$$\|s\| \leq \delta_c .$$

If $\delta_c \geq \left\|J(x_c)^{-1} F(x_c)\right\|$, then the step attempted is the Newton step. Otherwise, for the locally constrained optimal step, it is

$$s = -(J(x_c)^T J(x_c) + \mu_c I)^{-1} J(x_c)^T F(x_c) \qquad (3.5.5)$$

for μ_c such that $\|s\|_2 \approx \delta_c$.

Example 3.5.1: (cf.[3]) Let F: $\mathbb{R}^2 \to \mathbb{R}^2$, $F(x) = \begin{bmatrix} x_1^2 + x_2^2 - 2 \\ e^{x_1 - 1} + x_2^3 - 2 \end{bmatrix}$,

which has the root $x_+ = (1,1)^T$, and let $x_0 = (2,0.5)^T$.

Define $f(x) := \frac{1}{2} F(x)^T F(x)$.

Then

$$J(x_0) = \begin{bmatrix} 4 & 1 \\ e & 0.75 \end{bmatrix}, \quad F(x_0) \approx \begin{bmatrix} 2.25 \\ 0.843 \end{bmatrix},$$

and

$$s_0^N = -J(x_0)^{-1} F(x_0) \approx \begin{bmatrix} -3.00 \\ 9.74 \end{bmatrix}.$$

Our line –search Algorithm 2.3.1 will calculate $x_+ = x_0 + \lambda_0 s_0^N$ starting with $\lambda_0 = 1$, decreasing λ_0 if necessary until $f(x_+) < f(x_0) + 10^4 \lambda_0 \nabla f(x_0)^T s_0^N$.

For $\lambda_0 = 1$,

$$x_+ = x_0 + s_0^N \approx \begin{bmatrix} -1.00 \\ 10.24 \end{bmatrix}, F(x_+) \approx \begin{bmatrix} 104 \\ 1071 \end{bmatrix},$$

so that the Newton step clearly is unsatisfactory. Therefore we reduce λ_0 by a quadratic backtrack, calculating

$$\lambda_1 = \frac{-\nabla f(x_0)^T s_0^N}{2[f(x_+) - f(x_0) - \nabla f(x)^T s_0^N]} . \qquad (3.5.6)$$

In this case, $f(x_+) \approx 5.79 \times 10^5$, $f(x_0) \approx 2.89$,

$\nabla f(x_0)^T s_0^N = -F(x_0)^T F(x_0) \approx -5.77$,

So that (3.1.6) gives $\lambda_1 \approx 4.99\times10^{-6}$.

Since $\lambda_1 < 0.1$, Algorithm 2.3.1 gives $\lambda_1 = 0.1$,

$$x_+ = x_0 + 0.1s_0^N \approx \begin{bmatrix}1.70\\1.47\end{bmatrix},\ F(x_+) \approx \begin{bmatrix}3.06\\3.21\end{bmatrix}.$$

This is still unsatisfactory, and so Algorithm 2.3.1 does a cubic backtrack, then a backtrack yields $\lambda_2 \approx 0.0659$. Since $\lambda_2 > \frac{1}{2}\lambda_1$, the algorithm sets

$$\lambda_2 = \tfrac{1}{2}\lambda_1 = 0.05, x_+ = x_0 + 0.05s_0^N \approx \begin{bmatrix}1.85\\0.987\end{bmatrix}, F(x_+) \approx \begin{bmatrix}2.40\\1.30\end{bmatrix}.$$

This point is still unsatisfactory, since $f(x_+) \approx 3.71 > f(x_0)$, so the algorithm does another cubic backtrack. It yields $\lambda_3 \cong 0.0116$, which is used since it is in the interval

$$\left[\frac{\lambda_2}{10}, \frac{\lambda_2}{2}\right] = [0.005, 0.025].$$

Now we have

$$x_+ = x_0 + 0.016s_0^N \approx \begin{bmatrix}1.965\\0.613\end{bmatrix},\ F(x_+) \approx \begin{bmatrix}2.238\\0.856\end{bmatrix}.$$

This point is satisfactory, since

$$f(x_+) \approx 2.87 < 2.89 \approx f(x_0) + 10^{-4}(0.0116)\nabla f(x_0)^T s_0^N.$$

So we set $x_1 = x_+$ and proceed to the next iteration.

It is interesting to follow this problem further. At the next iteration,

$$s_1^N = -J(x_1)^{-1}F(x_1) \approx \begin{bmatrix}-1.22\\2.07\end{bmatrix},$$

$$x_2^N = x_1 + s_1^N \approx \begin{bmatrix}0.750\\2.68\end{bmatrix}, \qquad F(x_2^N) \approx \begin{bmatrix}5.77\\18.13\end{bmatrix},$$

which is again unsatisfactory. However, the first backtrack step is successful. The Algorithm 2.3.1 computes $\lambda_1 = 0.0156$, and, since this is less than 0.1, it sets $\lambda_1 = 0.1$,

$$x_+ = x_1 + 0.1s_1^N \approx \begin{bmatrix}1.84\\0.820\end{bmatrix}, \qquad F(x_+) \approx \begin{bmatrix}2.07\\0.876\end{bmatrix},$$

and the step is accepted, since

$$f(x_+) \approx 2.53 < 2.87 \approx f(x_1) + 10^{-4}(0.1)\nabla f(x_1)^T s_1^N .$$

At the next iteration, we get

$$s_2^N = -J(x_2)^{-1}F(x_2) \approx \begin{bmatrix} -0.0756 \\ 0.0437 \end{bmatrix},$$

$$x_3^N = x_2 + s_2^N \approx \begin{bmatrix} 1.088 \\ 1.257 \end{bmatrix}, \qquad F(x_3^N) \approx \begin{bmatrix} 0.762 \\ 1.077 \end{bmatrix},$$

and so the Newton step is very good . From here on, Newton's method converges q-quadratically to $x_* = (1,1)^T$.

References

[1] Danilin,Yu.M.\ Pshenichry, B.N: Numerical Methods in Extermal Problems, Mir Publishers, Moscow, 1978

[2] Deumlich, R: Optimization and Theory of Approximation, Textbook for Lecture, Addis Ababa, 1997.

[3] Dennis, J.E.\ Schnabel, B: Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice Hall, Eglewood Cliffs, 1983.

[4] Fletcher, R: Practical Methods of Optimization Volume, Johnwiley & Sons, USA, 1980.

[5] Murray, W: Numerical Methods for Unconstrained Optimization, Acadamic Press, London, NewYork, 1972.

[6] Powell, M.J.D: Nonlinear Optimization 1981, NATO Conference Series, Acadamic Press, London, 1982.