Addis Ababa University

አዲስ፡አበባ፡ዩኒቨርሲቲ

SEEK WISDOM, ELEVATE YOUR INTELLECT AND SERVE HUMANITY !

TRUST REGION NEWTON WITH DOGLEG METHOD

Abaye Yeshitla Mekonnen

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCE
DEPARTMENT OF MATHEMATICS

A Project submitted in partial fulfillment of the requirement of the degree of master of science in mathematics

Advisor:Berhanu Guta(PhD.)

May, 2018

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY
DEPARTMENT OF MATHEMATICS

The undersigned hereby certify that they have read and recommend to the department of mathematics for acceptance of this project entitled **"Trust region Newton with Dogleg Method"** by **Abaye Yeshitla** in partial fulfillment of the requirements for the degree of Master of Science in Optimization.

Advisor: Dr. Berhanu Guta

Signature:____________________

Date____________________

Examiner 1: Dr.____________________

Signature:____________________

Date____________________

Examiner 2: Dr. ____________________

Signature:____________________

Date____________________

ADDIS ABABA UNIVERSITY
DEPARTMENT OF MATHEMATICS

Author: Abaye Yeshitla

Title: Trust region Newton with Dogleg Method

Department: Mathematics

Degree: M.Sc.

Convocation: May

Year: 2018

Permission is herewith granted to Addis Ababa University to circulate and to have copied for non-commercial purposes, at its discretion, the above title upon the request of individuals or institutions.

Abaye Yeshitla :
Signature:____________________

Date____________________

Contents

List of Figures

List of Tables

Acknowledgment

First of all, I would like to thank my adviser, Dr. Berhanu Guta, who was eager, dedication and cooperative to help in any difficulty I faced in this project works as well as who seriously gives a constrictive comments to prepare this paper in a good manner. And I would also like to thank my classmates who gave their constructive comments and who participated on correcting connotations involved in this project. Finally I would like to thank all my Instructors for their motivation and guidance during the past two years at Addis Ababa University, department of Mathematics.

Abstract

In this project, we propose a trust region dogleg method algorithms to solve a trust region subproblems arising form unconstrained optimization. The method can deal with by constricting a dogleg paths. The case when the Hessian B of quadratic models is positive definite. The philosophy and fundamental ideas of trust region algorithms are discussed and proved that the method is globally convergent and has a supper linear convergence rate. And then the final algorithm is programmed in MATLAB and implemented by taking appropriate test problem.
Key-words:- Trust region, dogleg path, dogleg method, convergence.

Introduction

Consider an unconstrained minimization problem the mathematical formulation is

$$\min_{x} f(x) \tag{1}$$

where $x \in \Re^n$ is a real vector with components $n \geq 1$ and $f: \Re^n \rightarrow \Re$ is a smooth function.

Most numerical methods for optimization are iterative. That is, starting from an initial guess x_1 construct a sequence of points x_1, x_2, ..., x_k, ... which converges to a desired solution.
In order to solve any optimization problem numerically, nowadays there are two general iterative methods those are line search methods and Trust-region methods. Line search algorithms are one of the basic methods for solving optimization problems. These algorithms employ first determine descent directions p_k (called step direction) and then decide how far to move along p_k to find a number $\alpha_k \geq 0$ (called step length) such that f attains the least possible value at $x_k + \alpha_k p_k$ i.e, $f(x_k + \alpha_k p_k)$ sufficiently smaller than $f(x_k)$. The step length α_k may be obtained by solving (exactly or approximating) the one dimensional problem

$$\min_{\alpha > 0} f(x_k + \alpha_k p_k)$$

If α^* is the solution of this one dimensional problem, then set $\alpha_k = \alpha^*$ so that $x_{k+1} = x_k + \alpha_k p_k$. [1] Examples of such methods are: The steepest (gradient) Descent, Newton and Quasi Newton Methods. [1] In this paper an alternative method of solving unconstrained optimization problems will be examined, namely the Trust region method.

Trust-region method is one of the most important numerical optimization methods in solving non linear programming (NLP) problems. It is worth noting that trust region methods have a wide range of applications within the fields of science, engineering and even the social sciences. It works in a way that first determine an approximation of f around the x_k denoted by m_k (called model of f at x_k) and use a minimizer of $m_k(x_k + p)$ to construct x_{k+1} usually, m_k is a quadratic (second order Taylor) approximation of f around x_k, $m_k(x_k + p) \approx f(x_k + p)$ where $\|p\|$ is sufficiently small i.e, $m_k(x_k + p) = f(x_k) + \nabla f^T(x_k)p + 1/2p^T Bp$ where B_k is the Hessian matrix of f (or its approximation) at x_k and then find an (approximate) minimizer of the model on the region to locate x_{k+1}. Unlike the line search methods, Trust-region method usually determines the step size before the improving direction.
Note that The most prevalent of line search algorithms are the Newton and

Quasi-Newton methods, which is widely used within the field of optimization due to their fast (quadratic) convergence properties, given that certain conditions are satisfied. The trust region algorithm is in fact a modification of such methods, in that it restricts the Newton step within the bounds of the trust region. [3] The major drawback to the trust region method is that in order to obtain in the step p_k a minimization problem subject to one constraint (known as the Trust region sub-problem) must be solved. This is not trivial computationally, especially if there are large number of variables.

There are different methods are available to find an approximate solution of the given trust region subproblem. Thus are dogleg method, sub space minimization method and Steihaugs method. But, in this paper we are going to discuss, the simple, low cost and an efficient implementation to solve the trust-region subproblem is the so-called dogleg method which was presented by Powell [5]. To find an approximate solution of the trust subproblem Powell used a path consisting of two line segments to approximate p. The first line segment runs from the origin to the Cauchy point (a minimizer generated by the steepest descent method) i.e; $p^C = -\frac{g^T g}{g^T B g} g$; the second line segment runs from the Cauchy point to the Newton point i.e, $p^B = -B^{-1} g$.

The **main objective** of this paper is to find an approximate solution of the given trust region sub problem when the Hessian matrix of the model function B_k is positive definite. First to find a trial step p_k called (dogleg steps) and then by substituting the dogleg steps in to the trust region algorithm to find a solution of the given unconstrained optimization problems, develop an algorithm for both single dogleg methods and double dogleg methods and programmed in MATLAB implemented on a test problem .

The paper is organized as follows chapter 1 some important theorems and line search methods are presented. The philosophy of trust region sub problems and algorithms, methods used to solve trust region sub problems such as single and double dogleg methods and the prove of algorithms are globally convergent and that the supper linear convergence rate is discussed in chapter 2. Numerical results and its implementation is discussed in chapter 3. And the conclusion of this paper presented in chapter 4.

Chapter 1

Preliminaries

Consider an unconstrained minimization problem the mathematical formulation is

$$\min_{x} f(x) \tag{1.1}$$

where $x \in \Re^n$ is a real vector with components $n \geq 1$ and $f : \Re^n \rightarrow \Re$ is twice continuously differentiable function.

1.1 Optimal Solution and Optimal Value:

■ Let f(x)be objective function and $S \subseteq \Re^n$ be its feasible set

- x^* is called optimal solution for a minimization problem

  $$\min_{s.t \quad x \in S} f(x)$$

  If $x^* \in S$ and $f(x^*) \leq f(x) \quad for \quad all \quad x \in S$. Such x^* is also called global minimizer of f on S.

- x^* is called a local minimizer of f on S if there is $\epsilon > 0$ such that $f(x^*) \leq f(x)$ for all $x \in N_\epsilon(x^*) \bigcap S$, where, $N_\epsilon(x_0) = x \in v \ |x - x_0\| < \epsilon$, $\epsilon \ \ nbh \ \ of \ \ x_0$.

■ Optimal value if $x^* \in S$ is an optimal solution, then $f(x^*)$ is called the optimal (objective) value.

1.1.1 Convex function

Definition 1. *A function f: $\Re^n \rightarrow \Re$ is said to be convex if for any x_1, x_2 $\in \Re^n$ and any $\lambda \in [0, 1]$, $f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)$.*

Theorem 1. *Let f: $\Re^n \rightarrow \Re$ be a convex function.*
If x_0 is local minimizer of f, then it is global minimizer of f.

Theorem 2. *Let f: $\Re^n \rightarrow \Re$ be differentiable.*

1. $f(y) \geq f(x) + < \nabla f(x), y - x >$ *for every* $x,\ y \in S$ *iff f is convex.*
2. *Let f be twice continuously differentiable. The Hessian matrix of f is positive semi definite at each $x \in S$ iff f is convex on S.*

Remark: An $n \times n$ matrix A is said to be positive definite means $x^T Ax > 0$ for all non zero $x \in \Re^n$.
The mathematical tool used to study minimizers of smooth functions is Taylors theorem. Because this theorem is central to our analysis throughout the paper

Theorem 3. *(Taylors Theorem).*
Suppose that f: $\Re^n \rightarrow \Re$ is continuously differentiable and that $p \in \Re$. Then we have that

$$f(x+p) = f(x) + \nabla f(x+tp)^T p$$

for some $t \in (0, 1)$. Moreover, if f is twice continuously differentiable, we have that

$$\nabla f(x+p) = \nabla f(x) + \int_0^1 \nabla^2 f(x+tp) p dt$$

and that

$$f(x+p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x+tp) p$$

for some $t \in$ (0, 1). [1]

1.1.2 Basic optimality conditions:

Optimality conditions are used to check whether a point x^* is a local / global optimal solution for optimization problems.

Theorem 4. *(First-Order Necessary Conditions).*
If x^* is a local minimizer and f is continuously differentiable in an open neighborhood of x^*, then $\nabla f(x^*) = 0$. [1]

Theorem 5. *(Second-Order Necessary Conditions).*
If x^* is a local minimizer of f and $\nabla^2 f$ exists and is continuous in an open neighborhood of x^*, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semi definite. [1]

Theorem 6. *(Second-Order Sufficient Conditions).*
Suppose that $\nabla^2 f$ *is continuous in an open neighborhood of* x^* *and that* $\nabla f(x^*) = 0$ *and* $\nabla^2 f(x^*)$ *is positive definite. Then* x^* *is a strict local minimizer of f. [1]*

Theorem 7. *When f is convex, any local minimizer* x^* *is a global minimizer of f. If in addition f is differentiable, then any stationary point* x^* *is a global minimizer of f. [1]*

1.2 Overview of iterative solution Approach

■Most numerical methods for optimization are iterative. That is, starting from an initial guess x_1 construct a sequence of points $x_1, x_2, ..., x_k, ...$ which converges to a desired solution.

Definition 2. *Consider a function f:* $\Re^n \to \Re$ *and a point* $x_k \in dom(f)$*.*
A nonzero vector $p_k \in \Re^n$ *is called a descent direction of f at* x_k *iff there is* $\delta > 0$ *such that* $f(x_k + \alpha p_k) < f(x_k)$ *for all* $\alpha \in (0, \delta)$*.*

- *If a descent direction* p_k *is found at iterate* x_k*, then* $x_{k+1} = x_k + \alpha_k p_k$*, for some suitable* $\alpha_k \geq 0$*.*
- *If so,* p_k *is called step direction and* α_k *is called step length at* x_k*.*
- *The difference in iterative solution methods for NLP mainly lies on the way they identify step direction* p_k *and step length* α_k*.*

Theorem 8. *If* $\nabla f(x_k)^T p_k < 0$*, then* p_k *is a descent direction of f at* x_k*.*
(The angle between p_k *and* $-\nabla f(x_k)$ *must be acute angle).*

1.2.1 Convergence and Rate of convergence

■Convergence ensure that the sequence of iterates x_k converges to a desired solution x^* (x^* is optimal solution, at least locally).

Theorem 9. *If there is some* $r \in (0, 1)$ *and some* $k_0 \geq 0$ *such that* $\|x_{k+1} - x^*\| \leq r\|x_k - x^*\|$ *for all* $k \geq k_0$*, then* $x_k \to x^*$*.*

Definition 3. *(Rate of Convergence)*
Let x_k *be a sequence in* $\Re^n$ *that converges to* x^**. The rate of convergence is said be*

1. *linear :- if there is a constant* $r \in (0, 1)$ *and* $k_0 \geq 0$ *such that* $\frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|} \leq r$ *for all* $k \geq k_0$*.*

2. *supper linear:- if* $\lim_{k\to\infty} \frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|} = 0.$

3. *quadratic :- if there is a constant* $M > 0$ *and* $k_0 \geq 0$ *such that* $\frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|^2} \leq M$ *for all* $k \geq k_0$.

1.2.2 Line search Methods

Idea: At each iterate point x_k,

- determine a descent direction p_k (called step direction)
- Decide how far to move along p_k: Find a number $\alpha_k \geq 0$ (called step length) such that f attains the least possible value at $x_k + \alpha_k p_k$; i.e, $f(x_k + \alpha_k p_k)$ sufficiently smaller than $f(x_k)$.
- The step length p_k may obtained by solving (exactly or approximately) the one-dimensional problem min $f(x_k + \alpha p_k)$, s.t $\alpha > 0$.

There are various line search methods based on the way of determining p_k
From those we will discuss some of which is important with my project works.

The Steepest descent method

Most of the line search algorithms are implemented so that they have a monotone decrease of objective function. This means that the direction p_k needs to be a descending direction. Therefore the most logical direction to move along would be the steepest descent direction.
The steepest descent method is a line search method in which the search direction at x_k is given by

$$p_k = -\nabla f(x_k)$$

One advantage of the steepest descent direction is that it requires calculation of the gradient ∇f_k but not of second derivatives.
Properties of Steepest Descent Method:

- It is convergent (if the function has a minima) starting from any x_1.
- However, the convergence is slow (linear rate) due to its zigagging nature: $\nabla f(x_{k+1}) \perp p_k$ at every consecutive iterate points.

In general, any descent direction one that makes an angle of strictly less than $\Pi/2$ radians with ∇f_k is guaranteed to produce a decrease in f, provided that the step length $\alpha_k > 0$.

Newton's Method

Another important search direction-perhaps the most important one of all- is the Newton direction. This direction is derived from the second-order Taylor series approximation to $f(x_k + p)$, which is

$$f(x_k + p) \approx f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla^2 f_k p = m_k(p).$$

Assuming for the moment that $\nabla^2 f_k$ is positive definite, we obtain the Newton direction by finding the vector p that minimizes $m_k(p)$. By simply setting the derivative of $m_k(p)$ to zero, we obtain the following explicit formula:

$$p_k^N = -(\nabla^2 f_k)^{-1} \nabla f_k$$

The Newton direction can be used in a line search method when ∇f_k is positive definite, for in this case we have

$$\nabla f_k^T p_k^N = -(p^N)^T \nabla^2 f_k p_k^N \leq -\alpha_k \|p_k\|^2$$

for some $\alpha_k > 0$. We have that $\nabla f_k^T p_k^N < 0$, so the Newton direction is a descent direction. Newtons Method is fast (if it is applicable); but it has global convergence property typically quadratic. Only if the Hessian $\nabla^2 f(x_k)$ is positive definite (PD) at each iterate.

Modified Newtons method

Newtons method may not be descent method.
When $\nabla^2 f_k$ is not positive definite, the Newton direction may not even be defined, since $(\nabla^2 f_k)^{-1}$ may not exist. Even when it is defined, it may not satisfy the descent property $\nabla f_k^T p_k^N < 0$, in which case it is unsuitable as a search direction. In these situations, line search methods modify the definition of p_k to make it satisfy the descent condition in order to mitigate the drawbacks of the Newtons Method, we may make the following modifications in Newtons method:

1. Modify the Hessian matrix $\nabla^2 f(x_k)$ (approximate it by $n \times n$ symmetric PD matrix B_k), if needed, so that $p_k = -B_k^{-1} \nabla f(x_k)$ is descent direction (called Newtons descent direction).

2. Make a line search (exactly or inexactly) along p_k to avoid overshot, i.e, to have $f(x_{k+1}) \leq f(x_k)$.

The major issue is Hessian modification to get a sufficiently (PD) B_k. We will consider two types of Hessian modifications:
I Eigenvalue modfication

- Choose a sufficiently small $\delta > 0$ (say, δ= 0.01) (for measure of +ve definiteness).
- Let $\nabla^2 f_k$ be the Hessian of f at x_k; performing the spectral decomposition of $\nabla^2 f_k$, let $\nabla^2 f_k = Q^T \Lambda Q$ where $Q^T Q = I$, and Λ is the diagonal matrix of eigenvalues of $\nabla^2 f_k$.
- If $\lambda_i \geq \delta$ for all eigenvalues (diagonal elements of Λ, then $\nabla^2 f_k$ is sufficiently positive definite). So, $B_k = \nabla^2 f_k = Q^T \Lambda Q$.
- If $\lambda_i < \delta$ for some diagonal elements (eigenvalues) of Λ, then replace each of such diagonal elements by δ; and let $B_k = Q^T \Lambda Q$ (where Λ is with modified positive diagonals) So, B_k is positive definite. Use this factorized form of B_k to compute p_k at step 3; i.e, $p_k = -(Q^T \Lambda^{-1} Q)\nabla f_k$

II Modified Choleskey factorization

- If $\nabla^2 f_k$ is PD, then we can obtain its Cholesky factorization: $\nabla^2 f_k = L^T L$ (L lower triangular) matrix with +ve diagonal elements). This is possible if and only if $\nabla^2 f_k$ is PD. So, apply the Cholesky factorization procedure on $\nabla^2 f_k$ to get $L^T L$.

  - If succeed, $B_k = \nabla^2 f_k = L^T L$ is PD.
  - If not succeed, modify (increase) some diagonal elements of $\nabla^2 f_k$ to make the factorization possible. This can be accomplished by the following procedure:
  Modified Choleskey Factorization Procedure: Input: Hessian $\nabla^2 f_k = (a_{ij})$ at iterate x_k (Step 2 in Modified Newton) Choose a sufficiently small $\delta > 0$ (say, $\delta = 0.01$).

$$
\begin{aligned}
&for \;\; j = 1:n \\
&Let \;\; d_{jj} = max\{a_{jj} - \sum_{k=1}^{j-1} L_{jk}^2, \delta\} \\
&L_{jj} = \sqrt{d_{jj}} \\
&for \;\; i = j+1:n \\
&L_{ij} = (a_{ij} - \sum_{k=1}^{j-1} L_{jk} L_{ik}) / L_{jj} \\
&end
\end{aligned}
$$

Lemma 1. ***(Kantorovich :)*** *Let $A \in R^{n\times n}$ be a Symmetric and Positive definite matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$ Then for any $x \in \Re^n$ the following inequality holds*

$$1 \leq \frac{(x^T Ax)(x^T A^{-1}x)}{(x^T x)^2} \leq \frac{(\lambda_1 + \lambda_n)^2}{4\lambda_1\lambda_n} \quad [2]$$

Lemma 2. *We denote by $p^C = -g\frac{\|g\|^2}{g^T Bg}$; $p^B = -B^{-1}g$; $\gamma* = \frac{\|p^C\|^2}{(p^B)^T p^C}$ then $\gamma* \leq 1$, moreover if p^B is not parallel to p^C then $\gamma* < 1$.*

Proof. By using Kantorovich lemma 1 we can prove:
$(p^B)^T P^C = \|g\|^2 \frac{g^T B^{-1} g}{g^T Bg}$, $(p^C)^2 = \frac{\|g\|^6}{(g^T Bg)^2}$

we have

$$
\begin{aligned}
\gamma* =& \frac{\|p^C\|^2}{(p^B)^T p^C} \\
=& \frac{\|g\|^4}{[(g^T Bg)(g^T B^{-1}g)]} \quad and \quad using \quad Kantorovich \quad lemma 1 \\
\leq& 1
\end{aligned}
$$

inequality the lemma in proved.

□

Chapter 2

Trust-region Newton with Dogleg Method

2.1 Important concepts

In most cases, the trust-region is defined as a spherical area of radius Δ_k in which the trust-region subproblem lies but it is worth noting that shape of the trust region can differ depending on the type of norm used, however for simplicity this paper will only consider the Euclidean norm.

2.2 Trust region sub problem

We will assume that the model m_k that is used at each iterate x_k is quadratic. Moreover, m_k is based on the Taylor-series expansion of f around x_k, which is

$$f(x_k + p) = f_k + g_k^T p + \frac{1}{2} p^T \nabla^2 f(x_k + tp)p \tag{2.1}$$

where $f_k = f(x_k)$ and $g_k = \nabla f(x_k)$, and t is some scalar in the interval (0, 1). By using an approximation B_k to the Hessian in the second-order term, m_k is defined as follows:

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \tag{2.2}$$

where B_k is some symmetric matrix. The difference between $m_k(p)$ and $f(x_k + p)$ is $O(\|p\|)^2$, which is small when p is small. When B_k is equal to the true Hessian $\nabla^2 f_k$, the approximation error in the model function m_k is $O(\|p\|)^3$, so this model is especially accurate when $\|p\|$ is small. This choice $B_k = \nabla^2 f(x_k)$ leads to the trust-region Newton method.

If the problem dimension is not too large, the choice $B_k = \nabla^2 f(x_k)$ is reasonable and leads to second order Taylor models (2.2), the method based on this choice of model function are called **Trust region Newton methods**.
▲ Trust region Newton methods are not simply the Newton method with an additional size restriction.
▲Trust region Newton is a descent method, where as this is not guaranteed for Newton.

If we are using the quadratic model to approximate the original objective function, then our optimization problem is essentially reduced to solving a sequence of trust-region sub problems.

$$\begin{array}{c} Min \;\; m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \\ S.t \;\; \|p\| \leq \mathit{\Delta}_k \end{array} \tag{2.3}$$

where $\Delta_k > 0$ is the trust-region radius. g_k is the gradient at current point and B_k is the Hessian (or a Hessian approximation). In most of our discussions, we define $\|.\|$ to be the Euclidean norm, so that the solution p_k^* of (2.3) is the minimizer of m_k in the ball of radius Δ_k. Thus, the trust-region approach requires us to solve a sequence of subproblems (2.3) in which the objective function and constraint (which can be written as $p^T p \leq \Delta_k^2$) are both quadratic. When B_k is positive definite and $\|B_k^{-1} g_k\| \leq \Delta_k$, the solution of (2.3) is easy to identify it is simply the unconstrained minimum $p_k^B = -B_k^{-1} g_k$ of the quadratic $m_k(p)$. In this case, we call p_k^B is the Newton full step. Then, choosing a trust-region radius Δ_k>0, minimizing (approximately) $m_k(p)$ over the trust-region.

$$\{p \in R^n \;\; | \;\; \|p\| \leq \Delta_k\}$$

So, if p_k is acceptable approximate solution of the sub problem (2.3) then $x_{k+1} = x_k + p_k$
Major issues,

1. How do we choose Δ_k ?
2. How do we get suitable solution p_k of (2.3) ?

2.2.1 Choice of trust radius Δ_k:

Choosing initially at x_1 a sufficiently small Δ_1 we update Δ_k at each subsequent iterations depending on whether or not there is a good argument between $m_k(p)$ and $f(x_k + p)$ within the trust region at previous step.
Note that: If $\mathit{\Delta}_k$ is sufficiently small, then $m_k(p)$ and $f(x_k + p)$ are agree over the trust region but this may hinder the progress of iterates (this calls

for largeing trust radius at the next iterate).
If Δ_k is large then $m_k(p)$ and $f(x_k + p)$ may not agree and a minimizer of $m_k(p)$ over the trust-region can fail to produce acceptable next iterate (this calls for shrinking trust radius at the next iterate).

2.2.2 Actual and predicted reduction

The most critical issue underlying the trust-region method is to update the size of the trust-region at every iteration. If the current iteration makes a satisfactory reduction, we may exploits our model more in the next iteration, by setting a larger Δ_k. If we only achieved a limited improvement after the current iteration, the radius of the trust-region by adjusting the radius to a smaller value to check the model's validity. To measure the argument between $m_k(p)$ and $f(x_{k+p})$ we define the ratio.

$$\rho_k = \frac{f(x_k) - f(x_k + p_k^*)}{m_k(0) - m_k(p_k^*)} \tag{2.4}$$

The numerator is called actual reduction, and the denominator is the predicted reduction. Where to take a more ambitious step or a more conservative one is defend on the ratio between the actual reduction gained by true reduction in the optional objective function and the predicted reduction expected in the model function.
Note that since the step p_k is obtained by minimizing the model m_k over a region that includes $p = 0$, the predicted reduction will always be nonnegative. Hence, if ρ_k is negative, the new objective value $f(x_k + p_k)$ is greater than the current value $f(x_k)$, so the step must be rejected. Empirical threshold values of the ratio p_k will guide us in determining the size of the trust-region.

- ρ_k near 1, say $\rho_k \in [\frac{3}{4}, 1]$, indicated $m_k(p)$ agrees well with $f(x_k + p)$. (If this a case when Δ_k restricted p_k^*, then enlarge the current trust radius at the next iterate say, $\Delta_{k+1} = 2\Delta_k$).
- ρ_k negative or very close to 0 indicates m_k does not agree with f. (In this case we shrink the trust region by reducing Δ_k at the next iteration)
- If ρ_k is positive but significantly smaller than 1, we do not alter the trust region.

2.2.3 Trust-region Algorithm

Before implementing the trust-region algorithm, we should first determine several parameters.
Δ_k is the upper bound of the size of the trust region. η_1, η_2 and η_3, t_1, t_2 are the threshold values for evaluating the goodness of the Quadratic model thus for determining the trust-region's size in the next iteration. A typical set for those values are $0 \leq \eta_1 \leq \eta_2, \eta_2 = 0.25$ and $\eta_3 = 0.75, t_1 = 0.25, t_2 = 2.0$
The following procedure summarizes the process (Given iterate x_k, to generate x_{k+1})
Intial: Choose sufficiently small Δ_0 and $\eta \in [0, \frac{1}{4})$
For k=1, 2, 3, ...,
Get p_k^* by solving (Approximately) (2.3)
Evaluate ρ_k using (2.4)

$$
\begin{aligned}
&if \quad \rho_k < \frac{1}{4} \\
&\qquad \Delta_{k+1} = 0.25\Delta_k \\
&else\,if \quad p_k \geq \frac{3}{4} \;\; and \;\; \|p_k^*\| = \Delta_k \\
&\qquad \Delta_{k+1} = 2\Delta_k \\
&else \\
&\qquad \Delta_{k+1} = \Delta_k \\
&end \quad \% \;\; if \\
&if \qquad \rho_k{>}\eta \\
&\qquad x_{k+1} = x_k + p_k^* \qquad (\% \quad p_k^* \;\; is \;\; accepted) \\
&else \\
&\qquad x_{k+1} = x_k \qquad\qquad (\%\; p_k^* \quad is \;\; rejected) \\
&end \quad \%for
\end{aligned}
$$

Here $\hat{\Delta}$ is an overall bound on the step lengths. Note that the radius is increased only if $\|p_k\|$ actually reaches the boundary of the trust region. If the step stays strictly inside the region, we infer that the current value of Δ_k is not interfering with the progress of the algorithm, so we leave its value unchanged for the next iteration.

2.3 Methods of solving the trust-region subproblem

We need to focus on solving the trust-region subproblem (2.3). In discussing this matter, we sometimes drop the iteration subscript k and restate the problem (2.3) as follows:

$$\begin{array}{c} Min \;\; m(p) = f + g^T p + \frac{1}{2} p^T B p \\ S.t \;\; \|p\| \leq \varDelta \end{array} \tag{2.5}$$

2.3.1 The exact solution

Trust region subproblem can be solved exactly or approximately. Newton-like method is an exact solution. This method is meant to be applied to very small dimensional problems due to its computational complexity.

Lemma 3. *The vector p^* is the solution of the subproblem (2.5) if and only if there is a scalar $\lambda \geq 0$ such that*

$$(B + \lambda I)p^* = -\,g \tag{2.6}$$
$$(B + \lambda I) \geq 0 \tag{2.7}$$
$$\|p^*\| \leq \triangle \tag{2.8}$$
$$\lambda(\Delta - \|p^*\|) = 0 \tag{2.9}$$

and $(B + \lambda I)$ is positive semidefinite.

Proof. ($\Rightarrow$) Let p^* be a solution of sub problem (2.5). From the optimality condition of constrained optimization, there exist a multiplier $\lambda \geq 0$ such that (2.6)-(2.9) hold. We now need to prove that the matrix $(B + \lambda I)$ is positive semidefinite.
case1 if $\|p^*\| < \Delta$ then $\lambda = 0$ and p^* is unconstrained minimizer of m, and thus B is positive semidefinite
case2 if $\|p^*\| = \Delta$, it follows from the second order necessarily condition that

$$p^T(B + \lambda I)p \geq 0 \tag{2.10}$$

for all p satisfying $p^T p^* = 0$. If $p^T p^* \neq 0$ take $t = -2p^T p^*/\|p\|^2$ then $\|p^* + tp\| = \Delta$.
By definition of p^*,

$$m(p^* + tp) + \frac{1}{2}\lambda\|p^* + tp\|^2 \geq m(p^*) + \frac{1}{2}\lambda\|p^*\|^2 \tag{2.11}$$

Developing m(.)yields

$$\frac{1}{2}t^2p^T(B+\lambda I)p \geq -t(p^T[g+(B+\lambda I)p^*]) \tag{2.12}$$

By using (2.6) we get the right hand side of (2.12)

$$\frac{1}{2}t^2p^T(B+\lambda I)p \geq -t(p^T[g-g])$$

$\Rightarrow \frac{1}{2}t^2p^T(B+\lambda I)p \geq 0$
$\Rightarrow p^T(B+\lambda I)p \geq 0$ for all p with. Therefore, $B+\lambda I$ is positive definite
($\Leftarrow$) Assume that there is, $\lambda \geq 0$ such that p^* satisfies (2.6)-(2.9) and that $B+\lambda I$ is positive semidefinate. Then,for all p satisfying $\|p\| \leq \Delta$ we have

$$\begin{aligned} m(p) =& f(x) + g^Tp + \frac{1}{2}p^T(B+\lambda I)p - \frac{1}{2}\lambda\|p\|^2 \\ \geq& f(x) + g^Tp^* + \frac{1}{2}(p^*)^T(B+\lambda I)p^* - \frac{1}{2}\|p\|^2 \\ =& m(p^*) + \frac{1}{2}\lambda[\|p^*\|^2 - \|p\|^2] \end{aligned}$$

By use of (2.9), we have that $\lambda(\Delta^2 - (p^*)^Tp^*) = 0$ so, the above inequality become

$$\begin{aligned} m(p) \geq& m(p^*) + \frac{1}{2}\lambda[(\|p^*\|^2 - \Delta^2) + (\Delta^2 - \|p\|^2)] \\ =& m(p^*) + \frac{1}{2}\lambda[\Delta^2 - \|p\|^2] \end{aligned}$$

Thus, from $\lambda \geq$ and $\|p\| \leq \Delta$, we immediately have

$$m(p) \geq m(p^*)$$

which implies p^* is a solution of (2.5) □

2.3.2 The approximate solution

When the dimension is large, factorization to solve (2.6)-(2.9) can be difficult. Exact solution works when $B_k + \lambda I$ is positive definite and the process is complicated to calculate all eigenvectors and eigenvalues of B_k to find the optimal trajectory. Therefore we consider approximate solutions for a hard trust-region problem. There are two main approaches: Cauchy point, Dogleg method. we describe strategies for finding approximate solutions of the subproblem (2.3), which achieve at least as much reduction in m_k as the reduction achieved by the so-called Cauchy point. This point is simply the minimizer of m_k along the steepest descent direction g_k. Subject to the trust-region bound. The first approximate strategy is the dogleg method, which is appropriate when the model Hessian B_k is positive definite.

2.3.3 Cauchy point calculation

In line search methods, we may find an implementing direction from the gradient information, that is, by taking the steepest descent direction with regard to the maximum range we could make, we can solve the trust region subproblem in an inexpensive way. This method is also denoted as the Cauchy point calculation. we can also express the improving step explicitly by the following closed form equation.
Let the Cauchy point, denoted by p^c, is the minimizer of $m_k(p)$ s.t $\|p\| \leq \mathit{\Delta}_k$ in the direction of $-\nabla f_k^T$. So, if p_k^s is the point on the boundary of the trust region in the direction of $-\nabla f_k^T$, then

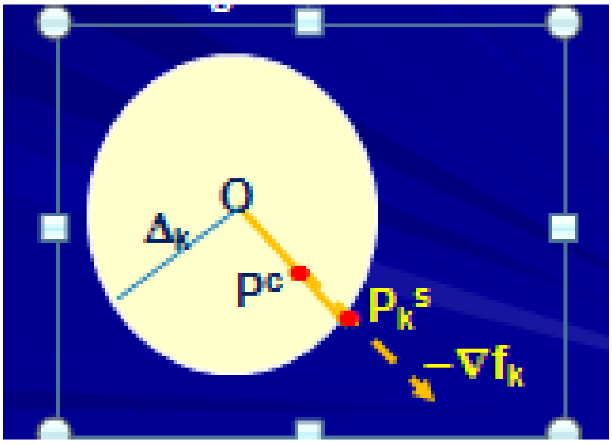

Figure 2.1: cauchy point

$p^c = \tau_k p_k^s$ some $\tau_k \in (0, 1]$
Find the vector p_k^s that solves a linear version of (2.3),

$$\begin{array}{c} Min \quad m(p) = f + g^T p \\ S.t \quad \|p\| \leq \mathit{\Delta} \end{array} \tag{2.13}$$

we get, $p_k^s = \frac{-\Delta_k g_k}{\|g_k\|} \Rightarrow p^c = \tau_k p_k^s = -\tau_k \Delta_k \frac{g_k}{\|g_k\|}, 0 < \tau_k \leq 1$
where, $\tau_k \in (0, 1]$ is the minimizer of $\phi(\tau) = m_k(\tau p_k^s) \quad S.t \quad \tau \in (0, 1]$

$\Rightarrow \phi(\tau) = m_k(\tau p_k^s) = f_k + g_k^T(\tau p_k^s) + \frac{1}{2}(\tau p_k^s)^T B_k(\tau p_k^s)$

$= f_k + \tau g_k^T p_k^s + \frac{1}{2}\tau^2 (p_k^s)^T B_k P_k^s$

$\Rightarrow \phi'(\tau) = g_k^T p_k^s + \tau (p_k^s)^T B_k p_k^s$

$= g_k^T(\frac{-\Delta_k}{\|g_k\|} g_k) + \tau(\frac{-\Delta_k}{\|g_k\|} g_k)^T B_k(\frac{-\Delta_k}{\|g_k\|} g_k)$

$=-\Delta_k\|g_k\| + \tau(\frac{\Delta_k}{\|g_k\|})^2 g_k^T B_k g_k$

$$\Rightarrow \phi'(\tau) = -\Delta_k\|g_k\| + \tau(\frac{\Delta_k)}{\|g_k\|})^2 g_k^T B_k g_k$$

To obtain τ_k explicitly, we consider the cases of $g_k^T B_k g_k \leq 0$ and $g_k^T B_k g_k > 0$ separately

- If $g_k^T B_k g_k \leq 0$ the function $m_k(\tau p_k^S)$ decreases monotonically with τ whenever $g_k \neq 0$, so τ_k is simply the largest value that satisfies the trust-region bound, namely, $\tau_k = 1$. (i.e $\phi'(\tau)<0 \ \ \forall \tau$ this implies that $\tau_k = 1$)
  Then $p^c = \frac{-\Delta_k}{\|g_k\|} g_k$ when $g_k^T B_k g_k \leq 0$

- If $g_k^T B_k g_k>0$, $m_k(\tau p_k^S)$ is a convex and quadratic in τ, then the above equation $\phi'(\tau) = 0$

  $\Rightarrow \quad -\Delta_k\|g_k\| + \tau_k(\frac{\Delta_k^2}{\|g_k\|^2}) g_k^T B_k g_k = 0$

  $\Rightarrow \quad \tau_k(\frac{\Delta_k^2}{\|g_k\|^2}) g_k^T B_k g_k = \Delta_k\|g_k\|$

  $\Rightarrow \quad \tau_k = \frac{\Delta_k\|g_k\|^3}{\Delta_k^2 g_k^T B_k g_k}$

  $\Rightarrow \quad \tau_k = \frac{\|g_k\|^3}{\Delta_k g_k^T B_k g_k}$
  $\tau_k = min\{\frac{\|g_k\|^3}{\Delta_k g_k^T B_k g_k}, 1\}$ when $g_k^T B_k g_k > 0$
  Summary: The Cauchy point is $p^c = \frac{-\tau_k \Delta_k g_k}{\|g_k\|}$
  Where,

  $$\tau_k = \begin{cases} 1 & if \quad g_k^T B_k g_k \leq 0 \\ min \ \{\frac{\|g_k\|^3}{\Delta_k g_k^T B_k g_k}, 1\} & otherwise \end{cases} \tag{2.14}$$

  Note:

- If $\tau = 1$ i.e, $\|p^c\| = \Delta_k$, then we take $p_k^* = p^c$ since the cauchy point provides sufficient reduction in model function and the iterate can progress at most up to boundary of the trust region.

- If $\tau_k < 1$ i.e, $\|p^C\| \leq \Delta_k$ then we consider improving the cauchy point by allowing B_k to play role in determining step direction (in order to improve the rate of convergence) such improvement is possible only if B_k is positive definite.

- So, If B_k is not positive definite, then we take $p_k^* = p^c$ and terminate the search for p_k^*
- How ever, If $\tau_k < 1$ and B_k is positive definite, then we can improve the cauchy point using the so called dogleg method.

2.3.4 Limitations and Improvements on the Cauchy point

Since the Cauchy point p_k^C provides sufficient reduction in the model function m_k to yield global convergence, and since the cost of calculating it is so small, **why should we look any further for a better approximate solution of (2.3)?** The reason is that:

- By always taking the Cauchy point as our step, we are simply implementing the steepest descent method with a particular choice of step length so it has only linear rate of convergence.
- The problem with using the Cauchy-point at each iteration is that it is always in the steepest descent direction and therefore, performs poorly in terms of speed.
- The Cauchy point does not depend very strongly on the matrix B_k, which is used only in the calculation of the step length. Rapid convergence can be expected only if B_k plays a role in determining the direction of the step as well as its length, and if B_k contains valid curvature information about the function.

A number of trust-region algorithms compute the Cauchy point and then try to improve on it. The improvement strategy is often designed so that the full step $p_k^B = -B_k^{-1} g_k$ is chosen whenever B_k is positive definite and $\|p_k^B\| \leq \Delta_k$. When B_k is the exact Hessian $\nabla^2 f(x_k)$ or a quasi-Newton approximation, this strategy can be expected to yield super linear convergence. A method that is currently being used as an improvement to the Cauchy point approach is the dogleg method. Throughout this section we will be focusing on the internal workings of a single iteration, so we simplify the notation by dropping the subscript k from the quantities Δ_k, p_k, m_k, and g_k and refer to the formulation (2.5) of the subproblem. In this section, we denote the solution of (2.5) by $p^*(\Delta)$, to emphasize the dependence on Δ.

2.4 Dogleg method

An efficient implementation to solve the trust-region subproblem is the so-called dogleg method which was presented by Powell [6]. This method finds an approximate solution of the subproblem (2.5), i.e, to find $x_{k+1} = x_k + p_k$ such that $\|p_k\| = \Delta_k$, Dogleg method is a combination of cauchy point (or steepest descent) and Newton point. When B is positive definite. To incorporate fast quadratic convergence rate offered by full Newton step and global convergence feature of steepest descent into a single trust region Algorithm an approach say a dogleg method is an efficient for solving the sub-problem. This method finds a compromise between steepest descent step and Newtons step based on the size of the trust region.
Note that

- If $\tau_k < 1$ i.e $\|p^C\| \leq \Delta_k$ then we consider improving the cauchy point by allowing B_k to play role in determining step direction (in order to improve the rate of convergence) such improvement is possible only if B_k is positive definite.
- So, If B_k is not positive definite, then we take $p_k^* = p^c$ and terminate the search for p_k^*.
- However, If $\tau_k < 1$ and B_k is positive definite, then we can improve the cauchy point using the so called dogleg method.

Remark: Dogleg method is an efficient and cheap to compute but it works only B_k is positive definite. Otherwise, we can define p^B choosing B to be one of the positive definite modified Hessian described in the preliminaries section.
To motivate this method, we start by examining the effect of the trust-region radius Δ on the solution $p^*(\Delta)$ of the subproblem (2.5). When B is positive definite, we have that the unconstrained minimizer of m is $p^B = -B^{-1}g$. Is determine by

$$\begin{aligned} m(p) =& f_k + g^T P + \frac{1}{2} p^T B p \\ \Rightarrow& \nabla m(p) = 0 \quad since \quad B \quad is \quad positive \quad definate \\ \Rightarrow& g + pB = 0 \\ \Rightarrow& P^B = -B^{-1} g \qquad is \quad Newton \quad step \end{aligned}$$

$$p^B = -B^{-1} g \tag{2.15}$$

■ When this point is $\|p^B\| \leq \Delta$, it is obviously a solution, so we have

$$p^*(\Delta) = p^B, \tag{2.16}$$

■ When Cauchy point p^C is outside the trust region or at the boundary, i.e. $\|p^C\| \geq \Delta$ the approximate solution is located at the truncated Cauchy point p_k^C where τ_k is 1. Is writing

$$p^*(\Delta) = -\frac{\Delta g}{\|g\|}, \tag{2.17}$$

■ **When** $\|p^C\| < \Delta$ **and** $\|p^B\| > \Delta$, the dogleg method finds an approximate solution by replacing the curved trajectory for $p(\Delta)$ with a path consisting of two line segments. The first line segment runs from the origin to the Cauchy point (a minimizer cauchy point generated by the steepest descent method);

$$p^C = -\frac{g^T g}{g^T B g} g \tag{2.18}$$

while the second line segment runs from the Cauchy point p^C to the Newton

point p^B. Formally, we denote this trajectory by $\tilde{p}(\tau)$ for $\tau \in [0, 2]$, where

$$\tilde{p}(\tau) = \begin{cases} \tau p^C & 0 \leq \tau \leq 1 \\ p^C + (\tau - 1)(p^B - p^C) & 1 \leq \tau \leq 2. \end{cases} \tag{2.19}$$

The dogleg method chooses p to minimize the model m along this path, subject to the trust-region bound. Because the path $\tilde{p}(\tau)$ intersects the trust- region boundary $\|p\| = \Delta$ at exactly one point if $\|p^B\| \geq \Delta$, and no where otherwise. Since $m(p)$ is decreasing along the path, the chosen value of p will be at p^B of $\|p^B\| \leq \Delta$, otherwise at the point of intersection of the dogleg and trust-region boundary.

The single dogleg curve is from the central point x_k along the steepest descent to the Cauchy Point p_k^C and continues along a straight line segment to the Newton point p^B(see Figure 2.2). This curve looks like a dogleg and approximates the curved optimal trajectory (shown dashed). Then the method chooses x_{k+1} to be the point on this polygonal arc such that $\|p_k\| = \|x_{k+1} - x_k\| = \Delta_k$, unless $\|p^B\| = \|B^{-1} g_k\| \leq \Delta_k$, in which case x_{k+1} is the Newton point. If the curve is chosen in such a way that the distance from the origin x_k to p^C increases monotonically as one moves along the piecewise linear curve, then for any $\|P^B\| \geq \Delta_k$, there is a unique point x_{k+1} on the

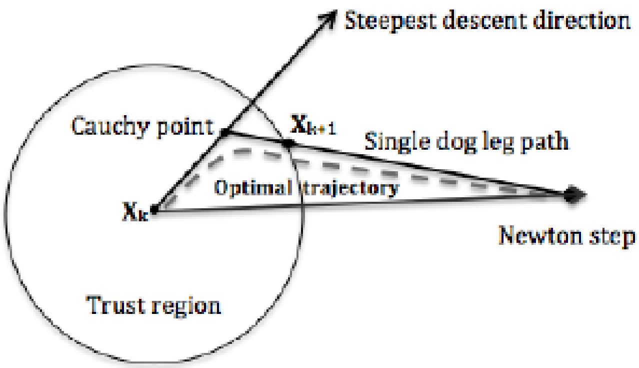

Figure 2.2: Single Dogleg path

curve such that $\|x_{k+1} - x_k\| = \Delta_k$. [9] This makes the process well defined. The following lemma shows the reason for finding the minimizer of the model along the dogleg curve.

Lemma 4. *let B be positive definite. Then*

1. *$\|\tilde{p}(\tau)\|$ is an increasing function of τ.*
2. *$m(\tilde{p}(\tau))$ is decreasing function of τ.*

Proof. First to show it holds true for $\tau \in [0,1]$ from the piece wise path:

$$\begin{aligned}
\phi(\tau) = \|\tilde{p(\tau)}\| =& \|\tau p^C\| \\
=& \| - \tau \frac{gg^T}{g^T B g} g\| \\
=& \| - \tau \frac{\|g\|^2}{g^T B g} g\| \\
=& \tau \|g\|^2 \frac{\|g\|}{\|g^T B g\|} \\
\phi'(\tau) =& \|g\|^2 \frac{\|g\|}{\|g^T B g\|} \\
\geq& 0 \forall \tau
\end{aligned}$$

Next our attention to the case of $\tau \in [1,2]$, for (1), define $h(\alpha)$ by

$$
\begin{aligned}
h(\alpha) =&\frac{1}{2}\|\tilde{p}(1+\alpha)\|^2 \\
=&\frac{1}{2}\|p^C + \alpha(p^B - p^C)\|^2 \\
=&\frac{1}{2}\|p^C\|^2 + \alpha(p^C)^T(p^B - p^C) + \frac{1}{2}\alpha^2\|p^B - p^C\|^2
\end{aligned}
$$

Our result is proved if we can show that $h'(\alpha) \geq 0 \quad for \quad \alpha \in (0,1)$. Now,

$$
\begin{aligned}
h'(\alpha) =& -(p^C)^T(p^C - p^B) + \alpha\|p^C - p^B\|^2 \\
\geq& -(p^C)^T(p^C - p^B) \\
=&\frac{g^T g}{g^T B g} g^T(\frac{-g^T g}{g^T B g} g + B^{-1} g) \\
=&\frac{g^T g}{g^T B g}(\frac{-g^T g}{g^T B g} g^T g + g^T B^{-1} g) \\
=&g^T g \frac{g^T B^{-1} g}{g^T B g}[1 - \frac{(g^T g)^2}{(g^T B g)(g^T B^{-1} g)}]
\end{aligned}
$$

$$
since\ by\ \ using\ \ Kantorovich\ \ lemma:
$$

$$
\gamma * = \frac{(g^T g)^2}{(g^T B g)(g^T B^{-1} g)} then\ \gamma * \leq 1
$$

$$
\begin{aligned}
=&g^T g \frac{g^T B^{-1} g}{g^T B g}[1 - \gamma *] \\
\geq&0
\end{aligned}
$$

□

Proof. First to show it holds true for $\tau \in [0,1]$

$$
\begin{aligned}
\phi(\tau) =& m(\tilde{p}(\tau)) = f_k + g^T(\tilde{p}(\tau)) + \frac{1}{2}(\tilde{p}(\tau))^T B(\tilde{p}(\tau)) \\
=& f_k + g^T(\tau p^C) + \frac{1}{2}(\tau p^C)^T B(\tau p^C) \\
=& f_k - \tau g^T(\frac{\|g\|^2}{g^T B g} g) + \frac{1}{2}\tau^2 \frac{(\|g\|^2)}{g^T B g} g^T B \frac{(\|g\|^2)}{g^T B g} g \\
\phi'(\tau) =& - g^T(\frac{\|g\|^4}{g^T B g} g) + 2\tau \frac{1}{2}\frac{(\|g\|^2)}{g^T B g} g^T B \frac{(\|g\|^2)}{g^T B g} g \\
=& - \frac{\|g\|^4}{g^T B g} + \tau \frac{\|g\|^4}{g^T B g} \qquad \textit{since} \quad \tau \leq 1 \\
\leq& 0
\end{aligned}
$$

Next case restrict our attention to the case of $\tau \in [1, 2]$ we define $\hat{h}(\alpha) = m(\tilde{p}(1+\alpha))$ and show that $\hat{h}(\alpha) \leq 0$ for $\alpha \in (0, 1)$. Substitution of (2.19) in to (2.5) and differentiation with respect to the argument leads to

$$
\begin{aligned}
\hat{h}'(\alpha) =& (p^B - p^C)^T(g + Bp^C) + \alpha(p^B - p^C)^T B(p^B - p^C) \\
=& (p^B - p^C)^T((g + Bp^C) + \alpha B(p^B - p^C)) \quad \textit{since} \ \alpha < 1 \\
\leq& (p^B - p^C)^T(g + Bp^C + B(p^B - p^C)) \\
=& (p^B - p^C)^T(g + Bp^B) \\
=& (p^B - p^C)^T(g + B(-B^{-1}g)) \\
=& (p^B - p^C)^T(0) \\
=& 0
\end{aligned}
$$

giving the result. □

It follows from this **lemma** that the path $\tilde{p}(\tau)$ intersects the trust- region boundary $\|p\| = \Delta$ at exactly one point if $\|p^B\| \geq \Delta$, and no where otherwise. Since $m(p)$ is decreasing along the path, the chosen value of p will be at p^B of $\|p^B\| \leq \Delta$, otherwise at the point of intersection of the dogleg and trust-region boundary. In the latter case, we compute the approximate value of α by solving the following quadratic equation:

$$
\begin{aligned}
\Delta =&\|p^C + \alpha(p^B - p^C)\| \\
\Rightarrow&\|p^C + \alpha(p^B - p^C)\|^2 = \Delta^2 \\
\Rightarrow&< p^C + \alpha(p^B - p^C), p^C + \alpha(p^B - p^C) >= \Delta^2 \\
&Let \quad c = p^c; b = p^B \\
\Rightarrow&< c + \alpha(b - c), c + \alpha(b - c) >= \Delta^2 \\
\Rightarrow&\alpha^2\|b - c\|^2 + 2\alpha c^T(b - c) + \|c\|^2 - \Delta^2 = 0 \\
\Rightarrow&\alpha^2\|c - b\|^2 + 2\alpha c^T(b - c) - (\Delta^2 - \|c\|^2) = 0 \\
\alpha =&\frac{-2c^T(b - c) \pm \sqrt{4(c^T(b - c))^2 + 4\|c - b\|^2(\Delta^2 - \|c\|^2)}}{2\|c - b\|^2} \\
\alpha =&\frac{c^T(c - b) \pm \sqrt{(c^T(c - b))^2 + \|c - b\|^2(\Delta^2 - \|c\|^2)}}{\|c - b\|^2}
\end{aligned}
$$

Take α only the positive roots because $\alpha \in (0, 1)$
Summery: if B_k is positive definite, $\|c\| < \Delta$ and $\|b\| > \Delta$ then $p^* = c + \alpha(b - c)$,
where α is given by above

2.4.1 Single dogleg method Algorithm

The approximate solution of the constrained minimization can be obtained by this simple and interesting algorithm

$$
\begin{aligned}
Step \quad 1&compute \quad \|p_k^C\| \quad by \quad (2.18); \\
&if \quad \|p_k^C\| \geq \Delta_k, \\
&p_k = -\frac{\Delta_k g_k}{\|g_k\|}; \\
&elseif \\
steep \quad 2 :&compute \quad \|p_k^B\| \quad by \quad (2.15) \\
&if \quad \|p_k^B\| \leq \Delta_k \\
&p_k = p_k^B; \\
&else \\
steep \quad 3 :&compute \quad \alpha \quad satisifying \quad \|p^C + \alpha(p^B - p^C)\| = \Delta; \\
p_k =&p^C + \alpha(p^B - p^C); \\
&end
\end{aligned}
$$

Where α is computed above.
Summery: Combining the above steps yields

$$x_{k+1} = \begin{cases} x_k - \frac{\Delta_k}{\|g\|} g_k, & \text{when} \quad \|p_k^c\| \geq \Delta_k \\ x_k - B^{-1} g_k, & \text{when} \quad \|p_k^c\| < \Delta_k \quad and \quad \|p_k^B\| \leq \Delta_k \\ x_k + p^C + \alpha(p^B - p^C) \ , & \text{when} \quad \|p_k^c\| < \Delta_k \quad and \quad \|p_k^B\| > \Delta_k \end{cases}$$

General trust region method with single dogleg steps
steep 1 Given $\bar{\Delta} > 0$,$\Delta_0 \in (0, \bar{\Delta})$Initial guess x_0, and $\eta \in [0, 1/4)$
For k=1,2,3,..., and tolerance are given
steep 2 Evaluate p_k approximately by dogleg method
Consider $sg = -\frac{g^T g}{g^T B g} g = -\frac{\|g\|^2}{g^T B g} g$
$sn = -B^{-1} g$
if $\|sg\| \geq \Delta_k$,
$p_k = -\frac{\Delta_k g_k}{\|g_k\|}$
else if $\|sn\| \leq \Delta_k$
$p_k = sn$
else
$b = sg^T(sg - sn)$
$c = \|sg - sn\|^2$
$d = \Delta^2 - \|sg\|^2$
$\alpha = (b + \sqrt{b^2 + cd})/c$
$p_k = sg + \alpha(sn - sg)$
end
$x_{k+1} = x_k + p_k$
steep 3 $\rho_k = (f(x_k) - f(x_{k+1})/m_k(0) - m_k(p)$

$$
\begin{aligned}
&if \quad \rho_k < \frac{1}{4} \\
&\qquad \Delta_{k+1} = 0.25\Delta_k \\
&else\ if \quad p_k \geq \frac{3}{4} \ \ and \ \ \|p_k^*\| = \Delta_k \\
&\qquad \Delta_{k+1} = 2\Delta_k \\
&else \\
&\qquad \Delta_{k+1} = \Delta_k \\
&end \quad \%if \\
&if \quad \rho_k > \eta \\
&\qquad x_{k+1} = x_k + p_k^* \qquad (\% \quad p_k^* \ \ is \ \ accepted) \\
&else \\
&\qquad x_{k+1} = x_k \qquad\qquad (\%\ p_k^* \quad is \ \ rejected) \\
&end \quad \%for
\end{aligned}
$$

2.4.2 Double dogleg method

Double dogleg method is proposed by Dennis and Mei [6] and is a modified single dogleg algorithm. A new step $p_k^B\ new$ is selected in the Newton direction and replaces the Newton step P_k^B by p_k^B new. This new point can be found by solving

$$
\begin{cases}
p_k^B \quad new = \eta p_k^B \\
\eta = 0.8\gamma + 0.2, \\
\gamma = \frac{\|g\|^4}{(g^TBg)(g^TB^{-1}g)}
\end{cases}
\tag{2.20}
$$

Dennis and Mei [2] found that if the point generated by trust-region iteration is biased towards the Newton direction, the behavior of the algorithm will be further improved. Then we choose a new point $\tilde{N}$ on the Newton direction, and connect the Cauchy point p^C to $\tilde{N}$. The intersection point of the connection line and the trust-region boundary is taken as the new iterate $\bar{x}_{k+1}$ (see$\bar{x}_{k+1}$ in Figure 2.3).

We say $x_k \to p^C \to$ Newton point as single dogleg, and $x_k \to p^C \to \tilde{N} \to$ Newton point as double dogleg.

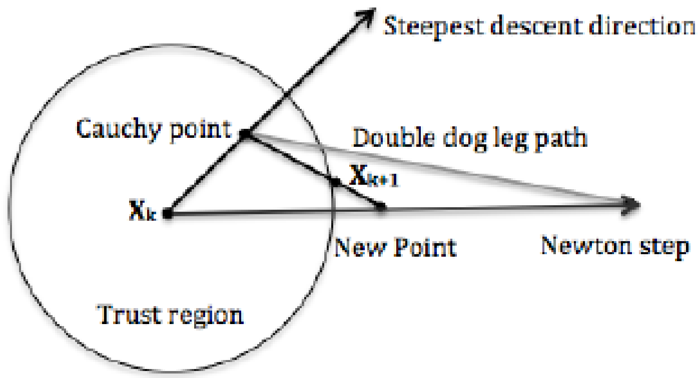

Figure 2.3: Double Dogleg path

In the Double dogleg approach we observe that the piecewise line connecting the Cauchy point p^C and the Newton point P^B can leave the trust region through two points instead of one point as shown in the diagram above. The points x_{k+1} and $\bar{x}_{k+1}$ are the points through which the path could leave the trust region, and if they are such that the model of the objective function decreases as we move along the path from x_k to p^B, then each of the points is a candidate for the next iterate whenever the constrained optimum lies outside the trust region, that is whenever $\|p^B\| \geq \Delta$.
Our proposal takes these two paths into consideration. One path involves two points, the points p^C and p^B while the other paths involves three points, the points $[p^C, \quad \tilde{N} = (\eta p^B) \quad and \quad p^B]$.
We can then approximate the curve either by two straight line segments or three line segments with the same end point. When B is positive definite, the double-dogleg method constructs a path with three line segments from the origin to the full step. The four points that define the path are

- the origin;
- the unconstrained Cauchy step $p^C = -\frac{g^T g}{g^T B g} g$;
- a fraction of the full step $\eta p^B = -\eta B^{-1} g$, for some $\eta \in (\gamma, 1]$,$\gamma = \frac{\|g\|^4}{[(g^T B g)(g^T B^{-1} g)]}$,
- the full step $p^B = B^{-1} g$.

described in the following form

$$\tilde{p}(\tau) = \begin{cases} \tau p^C, & 0 \leq \tau \leq 1 \\ p^C + (\tau - 1(\eta p^B - p^C)), & 1 \leq \tau \leq 2 \\ (\eta + (\tau - 2)(1 - \eta))p^B, & 2 \leq \tau \leq 3. \end{cases}$$

Show that $\tilde{p}^(\tau)$ increases monotonically along this path.
solution: The proof of this is the same procedure as the proof of the above lemma 3.
So $p(\tilde{\tau})$ is increases monotonically along this path. which is easy to proof.

Theorem 10. *In the double dogleg method, the distance from x_k to p^C, to $\tilde{N}$, is increasing monotonically.*

Proof. since

$$\begin{aligned}
\|p^C\| =& \| - \frac{\|g\|^2}{g^T B g} g\| \\
=& \frac{\|g\|^3}{(g^T B g)} \\
=& \frac{\|g\|^3}{(g^T B g)} \frac{(g^T B^{-1} g)}{(g^T B^{-1} g)} \\
\leq& \frac{\|g\|^3}{(g^T B g)} \frac{\|g\| \|B^{-1} g\|}{g^T B^{-1} g} \\
=& \frac{\|g\|^4}{(g^T B g)(g^T B^{-1} g)} \|p^B\| \\
& it \quad follows \quad from \quad Kantorovich \quad in \quad equality \\
\gamma =& \frac{\|g\|^4}{(g^T B g)(g^T B^{-1} g)} \leq 1 \\
\leq& \gamma \|p^B\| \\
Then \quad & \|p^C\| \leq \gamma \|p^B\| \leq \|p^B\| \\
Take \quad \tilde{N} \quad being \quad & x^N = x_k - \eta B_k^{-1} g_k = x_k + \eta p_k^B \quad where \quad \gamma \leq \eta \leq 1 \\
Thus \quad & \|x^C - x_k\| \leq \|x^N - x_k\| \leq \|x_{k+1}^N - x_k\|
\end{aligned}$$

$\Rightarrow$ which shows the property holds

□

Note: To develop the Algorithm of double dogleg method simply substitute (2.20) in to the dogleg algorithm above.

The approximate solution of the constrained minimization can be obtained by this simple algorithm.

$$
\begin{aligned}
& Step \quad 1 Compute \|p_k^C\| \quad by \quad (2.18); \\
& \qquad if \quad \|p_k^C\| \geq \Delta_k, \\
& \qquad p_k = -\frac{\Delta_k g_k}{\|g_k\|}; \\
& \qquad elseif \\
& steep \quad 2 Compute \quad \|p_k^B\| \quad by \quad (2.15) \\
& \qquad if \quad \|p_k^B\| \leq \Delta_k \\
& \qquad p_k = p_k^B; \\
& \qquad elseif \quad \|\eta p^B\| \leq \Delta_k \\
& \qquad where \quad \gamma = \frac{\|g\|^4}{(g^T B g)(g^T B^{-1} g)}, \eta = 0.8\gamma + 0.2, \\
& \qquad p_k = \Delta_k sn / \|sn\| \\
& \quad else \\
& steep \quad 3 Compute \quad \alpha \quad satisifying \quad \|p^C + \alpha(\eta p^B - p^C)\| = \Delta; \\
& \quad p_k = p^C + \alpha(\eta p^B - p^C); \\
& \qquad end
\end{aligned}
$$

Where α is computed above.
In summary: the double dogleg method choose the point$\tilde{N}$ which is defined by

$$x_{k+1} = x_k + \eta p_k^B, \eta \in [\gamma, 1] \tag{2.21}$$

When $\eta = 1$, the point $\tilde{N}$ is just the Newton point x_{k+1}^N and the double dogleg step is just the dogleg step. Generally, we take $\eta = 0.8\gamma + 0.2$. After generating the points C.P. and $\tilde{N}$, we find $x_{k+1}(\alpha) = x_k + p_k^C + \alpha(\eta p_k^B - p_k^C), 0 \leq \alpha \leq 1$. Such that

$$\|p_k^C + \alpha(\eta p_k^B - p_k^C)\|^2 = \Delta^2 \tag{2.22}$$

Now the value of α can be determined in the following way

$$\begin{aligned}
\Delta =&\|p^C + \alpha(\eta p^B - p^C)\| \\
\Rightarrow&\|p^C + \alpha(\eta p^B - p^C)\|^2 = \Delta^2 \\
\Rightarrow& < p^C + \alpha(\eta p^B - p^C), p^C + \alpha(\eta p^B - p^C) >= \Delta^2 \\
& Let \quad c = p^c; b = p^B \\
\Rightarrow& < c + \alpha(\eta b - c), c + \alpha(\eta b - c) >= \Delta^2 \\
\Rightarrow&\alpha^2\|\eta b - c\|^2 + 2\alpha c^T(\eta b - c) + \|c\|^2 - \Delta^2 = 0 \\
\Rightarrow&\alpha^2\|c - \eta b\|^2 + 2\alpha c^T(\eta b - c) - (\Delta^2 - \|c\|^2) = 0 \\
\alpha =& \frac{-2c^T(\eta b - c) \pm \sqrt{4(c^T(\eta b - c))^2 + 4\|c - \eta b\|^2(\Delta^2 - \|c\|^2)}}{2\|c - \eta b\|^2} \\
\alpha =& \frac{c^T(c - \eta b) \pm \sqrt{(c^T(c - \eta b))^2 + \|c - \eta b\|^2(\Delta^2 - \|c\|^2)}}{\|c - \eta b\|^2}
\end{aligned}$$

Thus double dogleg method could improve the convergence characteristics of the dog-leg method.
Consider now the case in which the exact Hessian $\nabla^2 f(x_k)$ is available for use in the model problem (2.5. When $\nabla^2 f(x_k)$ is positive definite, we can simply set $B = \nabla^2 f(x_k)$ (that is,$p^B = (\nabla f(x_k))^{-1} g_k$) and apply the procedure above to find the Newton-dogleg step. Otherwise, we can define p^B choosing B to be one of the positive definite modified Hessian described in the preliminaries section, then proceed as above to find the dogleg step. The use of a modified Hessian in the Newton-dogleg method is not completely satisfying from an intuitive viewpoint however, a modified factorization perturbs the diagonals of $\nabla^2 f(x_k)$ in a somewhat arbitrary manner, and the benefits of the trust-region approach may not be realized. In fact, the modification introduced during the factorization of the Hessian is redundant in some sense because the trust-region strategy introduces its own modification. As we observe in the exact solution of the trust-region problem (2.3) with $B_k = \nabla^2 f(x_k)$ is $(\nabla^2 f(x_k) + \lambda I)^{-1} g_k$, where λ is chosen large enough to make $(\nabla^2 f(x_k) + \lambda I)$ positive definite, and its value depends on the trust-region radius Δ_k. We conclude that the Newton-dogleg method is most appropriate when the objective function is convex (that is, $\nabla^2 f(x_k)$ is always positive semi definite).

2.5 Convergence of Trust-Region Methods

In the convergence analysis is important to obtain estimation of the reduction of the function to be minimized. A first step in this direction is the estimation

of the reduction of the model quadratic function.
In order to discuss the convergence of trust-region methods, we first give some assumptions and technical lemmas.
Assumptions listed below say (A_0)

- We assume that the approximate Hessian B_k are uniformly bounded in norm, and that f is bounded below on the level set

$$S = \{x | f(x) \leq f(x_0)\} \tag{2.23}$$

  is bounded, on which the function f: $R^n \to R$ is continuously differentiable. For later reference, we define an open neighborhood of this set by
  $S(R_0) = \{x | \|x - y\| < R_0 \quad for \quad some \quad y \quad \in S\}$,

- $\nabla^2 f(x)$ is Lipschitz continuous over S.

- we also allow the length of the approximate solution p_k of the subproblem (2.3) to exceed the trust-region bound, provided that it stays within a fixed multiple of the bound, that is

$$\|p_k\| \leq \gamma \Delta_k \tag{2.24}$$

  where γ is a positive constant

For trust-region algorithm, in general, we do not seek an accurate solution of subproblem (2.3) but we are content with a nearly optimal solution of (2.3). Our first main result is that the dogleg algorithm produce approximate solutions p_k of the subproblem (2.3) that satisfy the following estimate of decrease in the model function:

$$m_k(0) - m_k(p_k) \geq c_1 \|g_k\| min\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\} \tag{2.25}$$

where $c_l \in (0, 1]$. Below, we show that:

- The Cauchy point p_k^c satisfies (2.25) with $c_1 = \frac{1}{2}$

- The exact solution p_k of the subproblem (2.3) satisfies (2.25) with $c_1 = \frac{1}{2}$

- If p_k is an approximate solution of the subproblem (2.3) with $m_k(0) - m_k(p_k) \geq c_2(m_k(0) - m_k(p_k^c))$, then it satisfies (2.25)with $c_1 = \frac{c_2}{2}$.

Lemma 5. *The Cauchy point p_k^C satisfies*

$$m_k(0) - m_k(p_k^C) \geq \frac{1}{2}\|g_k\| min(\Delta_k, \frac{\|g_k\|}{\|B_k\|}) \tag{2.26}$$

Proof. Consider first the case of $g_k^T B_k g_k \leq 0$. In this case, it follows from (2.14) that $\tau_k = 1$,
and we have

$$\begin{aligned} m_k(0) - m_k(p_k^C) =& f_k - [f_k + g_k^T(p_k^C) + \frac{1}{2}(p_k^C)^T B_k p_K^C] \\ =& - g_k^T(-\frac{\Delta_k g_k}{\|g_k\|}) - \frac{1}{2}(-\frac{\Delta_k g_k}{\|g_k\|})^T B_k(-\frac{\Delta_k g_k}{\|g_k\|}) \\ =& \Delta_k\|g_k\| - \frac{1}{2}\frac{\Delta_k^2 g_k^T B_k g_k}{\|g_k\|^2} \\ \geq& \Delta_k\|g_k\| \\ \geq& \|g_k\| min\{\Delta_k, \frac{\|g_k\|}{\|B_k}\} \end{aligned}$$

Consider the case of $g_k^T B_k g_k > 0$ and

$$\frac{\|g_k\|^3}{(\Delta_k g_k^T B_k g_k)} \leq 1 \tag{2.27}$$

In this case $\tau_k = \frac{\|g_k\|^3}{(\Delta_k g_k^T B_k g_k)}$, and we have

$$\begin{aligned} m_k(0) - m_k(p_k^C) =& \frac{\|g_k\|^4}{g_k^T B_k g_k} - \frac{1}{2} g_k^T B_k g_k \frac{\|g_k\|^4}{(g_k^T B_k g_k)^2} \\ =& \frac{1}{2}\frac{\|g_k\|^4}{g_k^T B_k g_k} \\ \geq& \frac{1}{2}\frac{\|g_k\|^4}{\|g_k\|^2\|B_K\|} \quad \textit{By Cauchy schewartz inequailty} \\ =& \frac{1}{2}\frac{\|g_k\|^2}{\|B_k\|} \\ \geq& \frac{1}{2}\|g_k\| min\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\} \end{aligned}$$

Consider the case of $g_k^T B_k g_k > 0$ and

$$\frac{\|g_k\|^3}{(\Delta_k g_k^T B_k g_k)} > 1 \tag{2.28}$$

In this case $\tau_k = 1$, and by use of (2.28) we have

$$\begin{aligned}
m_k(0) - m_k(p_k^C) =& \Delta_k\|g_k\| - \frac{1}{2}\Delta_k^2 \frac{g_k^T B_k g_k}{\|g_k\|^2} \\
\geq& \Delta_k\|g_k\| - \frac{1}{2}\frac{\Delta_k^2}{\|g_k\|^2}\frac{\|g_k\|^3}{\Delta_k} \\
=& \frac{1}{2}\Delta_k\|g_k\| \\
\geq& \frac{1}{2}\|g_k\| min\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\}
\end{aligned}$$

The above discussion of three cases gives the result(2.25). □

Usually, we assume that p_k is an approximate solution of the subproblem (2.3) and satisfies

$$m_k(0) - m_k(p_k) \geq c_2(m_(0) - m_(p_k^e)) \tag{2.29}$$

where p_k^e is an exact solution of subproblem (2.3) and $c_2 \in (0, 1]$ is a constant. Since $m_k(p_k^e) \leq m_k(p_k^C)$, we immediately have,

$$m_k(0) - m_(p_k) \geq c_2(m_k(0) - m_k(p_k^C)) \tag{2.30}$$

where$p_k^C = -\tau_k \frac{\Delta_k}{\|g_k\|} g_k$ with $0 \leq \tau_k \leq 1$ is a Cauchy point. So, we immediately have

Lemma 6. *Let p_k be an approximate solution of (2.3) and satisfy (2.29)or (2.30). Then*

$$pred_k = m_k(0) - m_(p_k) \geq \frac{1}{2}c_2\|g_k\| min\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\} \tag{2.31}$$

where $c_2 \in (0, 1]$.

Proof. To proof this use (2.29),

$$\begin{aligned}
m_k(0) - m_(p_k) \geq& c_2(m_k(0) - m_k(p_k^C)) \\
\geq& c_2(\frac{1}{2}\|g_k\| min(\Delta_k, \frac{\|g_k\|}{\|B_k\|})) \quad By Lemma\ 3 \\
=& \frac{c_2}{2}(\|g_k\| min(\Delta_k, \frac{\|g_k\|}{\|B_k\|}))
\end{aligned}$$

The proof is completed □

Next, in order to prove the **global convergence theorem**, we give some technical lemmas.

Lemma 7. *Let Assumption* (A_0) *hold. We have*

$$|f(x_k + p_k) - m_k(p_k)| \leq \frac{1}{2}M\|P_k\|^2 + c(\|p_k\|)\|p_k\| \tag{2.32}$$

where $C(\|p_k\|)$ *is arbitrarily small by restricting the size of*p_k.

Proof. By Taylor's theorem,
$f(x_k + p_k) = f(x_k) + g_k^T p + \int_0^1 [\nabla f(x_k + tp_k) - \nabla f(x_k)]^T p_k dt$
$m_k(p_k) = f(x_k) + g_k^T p_k + \frac{1}{2}p_k^T B_k p_k$

$$\begin{aligned}
Then\ |f(x_k + p_k) - m_k(p_k)| =& |\frac{1}{2}p_k^T B_k p_k - \int_0^1 [\nabla f(x_k + tp_k) - \nabla f(x_k) p_k dt]| \\
\leq& |\frac{1}{2}p_k^T B_k p_k| + \int_0^1 |\nabla f(x_k + tp_k) - \nabla f(x_k)| p_k dt \ By\ Lipschitz\ contin \\
\leq& |\frac{1}{2}p_k^T B_k p_k| + \int_0^1 (x_k + tp_k - x_k) p_k dt \\
=& |\frac{1}{2}p_k^T B_k p_k| + \int_0^1 (tp_k) p_k dt \\
\leq& \frac{1}{2}M\|P_k\|^2 + c(\|p_k\|^2)
\end{aligned}$$

□

Theorem 11. *Let* $\eta = 0$ *in Algorithm 2.2.3 Suppose that* $\|B_k\| \leq \beta$ *for some constant* β*, that f is bounded below on the level set S defined by(2.23) and Lipschitz continuously differentiable in the neighborhood* $S(R_0)$ *for some* $R_0 > 0$*, and that all approximate solutions of (2.3) satisfy the inequalities (2.25) and (2.24), for some positive constants* c_1 *and* γ*. We then have*

$$\liminf_{k\to\infty} \|g_k\| = 0 \tag{2.33}$$

Proof. By performing some technical manipulation with the ratio ρ_k from (2.4), we obtain

$$\begin{aligned}
|\rho_k - 1| =& |\frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} - 1| \\
=& |\frac{f(x_k) - f(x_k + p_k) - [m_k(0) - m_k(p_k)]}{m_k(0) - m_k(p_k)}| \\
=& |\frac{m_k(p_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)}|
\end{aligned}$$

since from taylor's theorem we have that
$f(x_k+p_k) = f(x_k)+g_k^T + \int_0^1 [\nabla f(x_k+tp_k) - \nabla f(x_k)]^T p_k dt$ for some $t \in (0,1)$
and
$m_k(p_k) = f(x_k) + g_k^T p_k + \frac{1}{2} p_k^T B_k p_k$ 'it follows from the definition (2.2)

$$
\begin{aligned}
Then\ |m_k(p_k) - f(x_k+p_k)| =& |\frac{1}{2} p_k^T B_k p_k - \int_0^1 [\nabla f(x_k+tp_k) - \nabla f(x_k)]^T p_k dt| \\
\leq& |\frac{1}{2} p_k^T B_k p_k| + \int_0^1 [|\nabla f(x_k+tp_k) - \nabla f(x_k)]^T p_k dt| \\
\leq& \frac{\beta}{2} \|P_k\|^2 + \beta_1 \|p_k\|^2 \qquad (2.34)
\end{aligned}
$$

where we have used β_1 to denote the Lipschitz constant for g on the set $S(R_0)$, and assumed that $\|p_k\| \leq R_0$ to ensure that x_k and $x_k + tp_k$ both lie in the set $S(R_0)$.
Suppose for contradiction that there is $\epsilon > 0$ and a positive index K such that

$$\|g_k\| \geq \epsilon, for \quad all \quad k \geq K \qquad (2.35)$$

From (2.25),we have for $k \geq K$ that

$$
\begin{aligned}
m_k(0) - m_k(p_k) \geq& c_1 \|g_k\| min\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\} \\
\geq& c_1 \epsilon min\{\Delta_k, \frac{\epsilon}{\beta}\} \qquad (2.36)
\end{aligned}
$$

using (2.36),(2.34) and the bound (2.24) we have

$$|\rho_k - 1| = \frac{\gamma^2 \Delta_k^2 (\frac{\beta}{2} + \beta_1)}{c_1 \epsilon min(\Delta_k, \frac{\epsilon}{\beta})} \qquad (2.37)$$

We now derive a bound on the right hand side that holds for all sufficiently small values of Δ_k , that is, for all $\Delta_k \leq \bar{\Delta}_k$, where $\bar{\Delta}_k$ is defined as follows:

$$\bar{\Delta} = min(\frac{1}{2} \frac{c_1 \epsilon}{\gamma^2 (\frac{\beta}{2} + \beta_1)}, \frac{R_o}{\gamma}) \qquad (2.38)$$

The $\frac{Ro}{\gamma}$ term in this definition ensures that the bound (2.34) is valid (because $\|p_k\| \leq \gamma \Delta_k \leq \gamma \bar{\Delta}_k \leq R_0$). Note that since $c_1 \leq 1$and $\gamma \geq 1$, we have $\bar{\Delta}_k \leq \frac{\epsilon}{\beta}$
The latter condition implies that for all $\Delta_k \epsilon [0, \bar{\Delta}]$, we have $min(\Delta_k, \frac{\epsilon}{\beta}) = \Delta_k$,

so from(2.37) and (2.38), we have

$$
\begin{aligned}
|\rho_k - 1| \leq & \frac{\gamma^2 \Delta_k^2 (\frac{\beta}{2} + \beta_1)}{c_1 \epsilon \Delta_k} \\
= & \frac{\gamma^2 \Delta_k (\frac{\beta}{2} + \beta_1)}{c_1 \epsilon} \\
\leq & \frac{\gamma^2 \bar{\Delta} (\frac{\beta}{2} + \beta_1)}{c_1 \epsilon} \\
= & \frac{\gamma^2 (\frac{\beta}{2} + \beta_1)}{c_1 \epsilon} \times \frac{1}{2} \frac{c_1 \epsilon}{\gamma^2 (\frac{\beta}{2} + \beta_1)} \\
\leq & \frac{1}{2}
\end{aligned}
$$

Therefore, $\rho_k \geq 1/2 > \frac{1}{4}$, and so by the workings of Algorithm (2.36), we have $\Delta_{k+1} \geq \Delta_k$ whenever Δ_k falls below the threshold $\bar{\Delta}$. It follows that reduction of Δ_k(by a factor of$\frac{1}{4}$) can occur in our algorithm only if

$$\Delta_k \geq \bar{\Delta}$$

and therefore we conclude that

$$\Delta_k \geq min(\Delta_k, \bar{\Delta}/4) for \;\; \forall \;\; k \geq K \tag{2.39}$$

Suppose now that there is an infinite subsequence K such that $\rho_k \geq 1/4$ for $k \in K$.
For $k \in K$ and $k \geq K$, we have from (2.36)

$$
\begin{aligned}
f(x_k) - f(x_k + 1) = & f(x_k) - f(x_k + p_k) \\
\geq & \frac{1}{4} [m_k(0) - m_k(p_k)] \\
\geq & \frac{1}{4} c_1 \epsilon min(\Delta_k, \epsilon/\beta)
\end{aligned}
$$

Since f is bounded below, it follows from this inequality that

$$\lim_{k \in K, k \to \infty} \Delta_k = 0$$

contradicting (2.39). Hence no such infinite subsequence K can exist, and we must have $\rho_k < 1/4$ for all k sufficiently large. In this case, Δ_k will eventually be multiplied by 1/4 at every iteration,
and we have $\lim_{k \to \infty} \Delta_k = 0$, which again contradicts (2.39). Hence, our original assertion (2.35) must be false, giving (2.33). $\square$

for the case $\eta > 0$, borrows much of the analysis from the proof above. .

Theorem 12. *Let $\eta \in (0, \frac{1}{4})$ in Algorithm 2.3.3 Suppose that $\|B_k\| \leq \beta$ for some constant β, that f is bounded below on the level set S defined by and (2.23) Lipschitz continuously differentiable in the neighborhood $S(R_0)$ for some $R_0 > 0$, and that all approximate solutions of (2.3) satisfy the inequalities (2.25) and (2.24), for some positive constants c_1 and γ. We then have*

$$\lim_{k \to \infty} \|g_k\| = 0 \tag{2.40}$$

Proof. We consider a particular positive index m with $g(x_m) \neq 0$. Using β_1 again to denote the Lipschitz constant for g on the set $S(R_0)$, we have

$$\|g(x) - g(x_m)\| \leq \beta_1 \|x - x_m\|$$

for all $x \in S(R_0)$. We now define the scalars and R to satisfy

$$\epsilon = \frac{1}{2}\|g_m\|, R = min(\epsilon/\beta_1, R_0)$$

Note that the ball

$$\mathbf{B}(x_m, R) = \{x | \|x - x_m\| \leq R\}$$

is contained in $S(R_0)$, so Lipschitz continuity of g holds inside $\mathbf{B}(x_m, R)$.We have

$$\begin{aligned} x \in \mathbf{B}(x_m, R) \Rightarrow \|g(x)\| = &|g_m + g(x) - g_m\| \\ \geq &\|g_m\| - \|g(x) - g_m\| \\ \geq &2\epsilon - \beta_1 \|x - x_m\| \\ \geq &2\epsilon - \beta_1 min(\epsilon/\beta_1, R_0) \\ = &2\epsilon - \beta_1(\epsilon/\beta_1) \\ = &\epsilon \end{aligned}$$

If the entire sequence $\{x_k\}_k \geq m$ stays inside the ball $\mathbf{B}(x_m, R)$, we would have $\|g_k\| \geq \epsilon > 0$ for all $k \geq m$. The reasoning in the proof of Theorem 11 can be used to show that this scenario does not occur. Therefore, the sequence $\{x_k\}_k \geq m$ eventually leaves $\mathbf{B}(x_m, R)$.
Let the index $l \geq m$ be such that $x_l + 1$ is the first iterate after x_m outside

$\mathbf{B}(x_m, R)$. Since $\|g_k\| \geq \epsilon$ for $k = m, m+1, ..., l$, we can use (2.36) to write

$$\begin{aligned} f(x_m) - f(x_l + 1) &= \sum_{k=m}^{l} f(x_k) - f(x_k + 1) \\ &\geq \sum_{k=m, x_k \neq x_{k+1}}^{l} \eta[m_k(0) - m_k(p_k)] \\ &\geq \sum_{k=m, x_k \neq x_{k+1}}^{l} \eta c_1 \epsilon min(\Delta_k, \frac{\epsilon}{\beta}) \end{aligned}$$

where we have limited the sum to the iterations k for which $x_k = x_{k+1}$, that is, those iterations on which a step was actually taken. If $\Delta_k \leq \epsilon/\beta$ for all $k = m, m+1, ..., l$, we have

$$f(x_m) - f(x_l + 1) \geq \eta c_1 \epsilon \sum_{k=m, x_k \neq x_{k+1}}^{l} \Delta_k \geq \eta c_1 \epsilon R = \eta c_1 \epsilon min(\frac{\epsilon}{\beta_1}, R_0) \quad (2.41)$$

Otherwise, we have $\Delta_k > \epsilon/\beta$ for some $k = m, m+1, ..., l$, and so

$$f(x_m) - f(x_l + 1) \geq \eta c_1 \epsilon \frac{\epsilon}{\beta} \quad (2.42)$$

Since the sequence $\{f(x_k)\}_{k=0}^{\infty}$ is decreasing and bounded below, we have that

$$f(x_k) \downarrow f^* \quad (2.43)$$

for some $f^* > -\infty$ Therefore, using (2.42) and (2.43), we can write

$$\begin{aligned} f(x_m) - f^* &\geq f(x_m) - f(x_{l+1}) \\ &\geq \eta c_1 \epsilon min(\frac{\epsilon}{\beta} \frac{\epsilon}{\beta_1}, R_0) \\ &= \frac{1}{2} \eta C_1 \|g_m\| min(\frac{\|g_m\|}{2\beta}, \frac{\|g_m\|}{2\beta_1}, R_0) \\ &> 0 \end{aligned}$$

Since $f(x_m) - f^* \downarrow 0$, we must have $g_m \to 0$ giving the result. □

Theorem 13. *Let f be twice Lipschitz continuously differentiable in a neighborhhod of a point x^* at which second-order sufficient conditions (Theorem 5) are satisfied. Suppose the sequence $\{x_k\}$ converges to x^* and that for all k sufficiently large, the trust-region algorithm based on (2.3) with $B_k = \nabla^2 f(x_k)$*

chooses steps p_k that satisfy the Cauchy-point-based model reduction criterion (2.25) and are asymptotically similar to Newton steps p_k^B whenever $\|p_k^B\| \leq 1/2\Delta_k$, that is

$$\|p_k - p_k^B\| = o(\|p_k^B\|) \tag{2.44}$$

Then the trust-region bound Δ_k becomes inactive for all k sufficiently large and the sequence $\{x_k\}$ converges super linearly to x^*.

Proof. We show that $\|p_k^B\| \leq 1/2\Delta_k$ and $\|p_k\| \leq \Delta_k$, for all sufficiently large k, so the near-optimal step p_k in (2.44) will eventually always be taken
We first seek a lower bound on the predicted reduction $m_k(0) - m_k(p_k)$ for all sufficiently large k. We assume that k is large enough that the $o(\|p_k^B\|$ term in (2.44) is less than $\|p_k^B\|$.
When $\|p_k^B\| \leq 1/2\Delta_k$ then we have that

$$\|p_k\| \leq \|p_k^B\| + o(\|p_k^B\|) \leq 2\|p_k^B\|$$

while if $\|p_k^B\| > 1/2\Delta_k$, we have

$$\|p_k\| \leq \Delta_k < 2\|p_k^B\|$$

In both cases then we have

$$\begin{aligned}
\|p_k\| \leq& 2\|p_k^B\| \\
=& 2\|\nabla^2 f_k^{-1} g_k\| \\
\leq& 2\|\nabla^2 f_k^{-1}\|\|g_k\| \\
\Rightarrow \|g_k\| \geq& \frac{1}{2}\frac{\|p_k\|}{\|\nabla^2 f_k^{-1}\|}
\end{aligned}$$

we have from (2.25) that

$$\begin{aligned}
m_k(0) - m_k(p_k) \geq& c_1\|g_k\| min(\Delta_k, \frac{\|g_k\|}{\|\nabla^2 f(x_k)\|}) \\
\geq& c_1 \frac{\|p_k\|}{2\|\nabla^2 f_k^{-1}\|} min(\|p_k\|, \frac{\|p_k\|}{2\|\nabla^2 f_k\|\|\nabla^2 f_k^{-1}\|}) \\
=& c_1 \frac{\|p_k\|^2}{4\|\nabla^2 f_k\|\|\nabla^2 f_k^{-1}\|^2})
\end{aligned}$$

Because $x_k \to x^*$, we use continuity of $\nabla^2 f(x)$ and positive definite ness of $\nabla^2 f(x^*)$ deduce that the following bound holds for all k sufficiently large:

$$\frac{c_1}{4\|\nabla^2 f^{-1}(x_k)\|^2\|\nabla^2 f(x_k)\|} \geq \frac{c_1}{8\|\nabla^2 f^{-1}(x^*)\|^2\|\nabla^2 f(x^*)\|} = c_3$$

where $c_3 > 0$, hence, we have

$$m_k(0) - m_k(p_k) \geq c_3 \|p_k\|^2 \tag{2.45}$$

for all sufficiently large k, by lipschiz continuity of $\nabla^2 f(x)$ near x^*, and using Taylors theorem, we have

$$\begin{aligned}
|(f(x_k) - f(x_k + p_k)) - (m_k(0) - m_k(p_k))| =& |m_k(p_k) - f(x_k + p_k)| \\
& |\frac{1}{2} p_k^T \nabla^2 f(x_k) p_k - \frac{1}{2} \int_0^1 p_k^T \nabla^2 f(x_k + tp_k) p_k dt| \\
\leq & \frac{L}{4} \|p_k\|^3
\end{aligned}$$

where $L > 0$ is the Lipschitz constant for $\nabla^2 f(x)$. Hence, by definition (2.4) of ρ_k, we have for sufficiently large k that

$$\begin{aligned}
|\rho_k - 1| =& |\frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} - 1| \\
=& |\frac{m_k(p_k) - f(x_k + p_k}{m_k(0) - m_k(p_k)}| \\
\leq & \frac{\frac{L}{4} \|p_k\|^3}{c_3 \|p_k\|^2} \\
=& \frac{L}{4c_3} \|p_k\| \\
\leq & \frac{L}{4c_3} \Delta_k
\end{aligned} \tag{2.46}$$

Now, the trust-region radius can be reduced only if $\rho_k < 1/4$ (or some other fixed number less than 1), so it is clear from (2.46) that the sequence $\{\Delta_k\}$ is bounded away from zero. Since $x_k \to x^*$, we have $\|p_k^B\| \to 0$ and therefore $\|p_k\| \to 0$ from (2.44). Hence, the trust-region bound is inactive for all k sufficiently large, and the bound $\|p_k^B\| \leq \frac{1}{2}\Delta_k$ is eventually always satisfied
To prove super linear convergence,from the definition of the Newton step and the optimality condition $\nabla f(x^*)$=0

we have that

$$\begin{aligned}
x_k + p_k^N - x^* =& x_k - \nabla^2 f_k^{-1} \nabla f_k \\
=& \nabla^2 f_k^{-1}(x_k - x^*) - (\nabla f_k - \nabla f(x^*) \\
since \;\; Taylor's \; theorem(Thm5) \; tells \; us \; that \\
\nabla f_k - \nabla f(x^*) =& \int_0^1 \nabla^2 f(x_k + t(x^* - x_k))(x_k - x^*) dt \\
\|\nabla^2 f_k(x_k - x^*) - (\nabla f_k - \nabla f(x^*))\| =& \| \int_0^1 [\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k))](x_k - x^*) dt\| \\
\leq& \int_0^1 \|[\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k))\|\|(x_k - x^*)\| dt \\
\leq& \int_0^1 L\|x_k - x^*\|^2 dt \\
= 1/2L\|x_k - x^*\|^2.
\end{aligned}$$

Where L is lipschitz constant now by using this result we obtain

$$\|x_k + p_k^B - x^*\| = o(\|x_k - x^*\|)$$

which $\Rightarrow$ that $\|p_k^B\| = o(\|x_k - x^*\|)$ therefore using (2.44), we have

$$\begin{aligned}
\|x_k + p_k - x^*\| \leq& \|x_k + p_k^B - x^*\| + \|p_k^B - p_k\| \\
=& o(\|x_k - x^*\|^2) + o(\|p_k^B\|) \\
=& o(\|x_k - x^*\|)
\end{aligned}$$

Thus proving the supper linear. $\square$

Chapter 3

Numerical Implementation

In this section, we present the numerical results for the dogleg algorithm and the double dogleg algorithm. As we presented in chapter 2, approximate solutions are widely used for trust region problem to find the step p_k are dogleg algorithm and double dogleg method because those are simple and cheap to compute. Here, we applied both methods for optimization step and compare their corresponding results.

Rosenbrock function is a popular test problem for optimization algorithms. It is also referred as Rosenbrocks valley or Rosenbrocks banana function because of its curved contours. The two-dimensional Rosenbrock function is widely used for numerical optimization problems. It has a global minimum at(1,1), where $f(x_1; x_2)$ is 0. The definition of the function is

$$f(x_1; x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \tag{3.1}$$

Now consider this function we develop matlab code for both methods. We are taking the same initial point and tolerance for both methods.

Example 1. *Consider this* $f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ *at* $x_0 = [1.2, 1]$ *and* $\Delta_0 = 1.5$*,* $tol = 10^{-5}$ $\delta = 0.2$*.*
To find approximate solution this function by developing MATLAB code for single dogleg method and double dogleg method, Newton method and steepest descent method
The first search begins by choosing a start point very close to the minimum point of function. Take the initial point x=[1.2,1]. Then we look at the optimization results (see Table 3.1) when dogleg algorithm is used.
Solution

Iteration 1

The algorithm start from the initial point $x_0 = (1.2, 1)$, $f_k = 19.4$. *The trust region is defined as the area inside the circle, which is centred at the starting point and has radius 1.5.*

$sg = -\frac{\|g\|^2}{g^T Bg} g = [-0.1407, 0.0585]^T$, $\|sg\| = 0.1524$, $sn = -B^{-1}g = [-0.0022, 0.4346]^T$, $\|sn\| = 0.4346$. *Now compare the value of* $\|sg\|$ *and* $\|sn\|$ *to* Δ_0 *then by using the above step to find* p_k

$\|sn\| < \Delta_0$ $p_0 = [-0.0022, 0.4346]^T$ *and* $\|sn\| = 0.1524$ $\rho_0 = 1$, $\|p_0\| = 0.1524$. *This shows that the quadratic model is a good relation to the objective function. From the algorithm.* $\rho_0 => 3/4$ *but* $\|p_0\| \neq \Delta_0$. *The trust region radius is the same for the next iteration i.e,* $\Delta_1 = \Delta_0$ *and observe that* $\rho_0 > 0.2$. *Which gives* $x_1 = x_0 + p_0 = [1.1978, 1.4346]^T$

Iteration 2

Start with $x_1 = [1.1978, 1.4346]^T$ $\Delta_1 = 1.5$, $sg = [-0.0003797, 0.0000208]^T$, $\|sg\| = 0.00003802$, $sn = [-0.1930, -0.4622]^T$, $\|sn\| = 0.5009$. *Now compare the value of* $\|sg\|$ *and* $\|sn\|$ *to* Δ_1 *then we select* $\|sn\| < \Delta_1 \Rightarrow p_1 = sn$ $\|sn\| = 0.5009$. *The ratio* $\rho_1 = -2.8978$. *This shows that the quadratic model and the objective function is not related so the new point is not accepted. we need to reduced the trust region radius* $\Delta_2 = 0.25\Delta_1 = 0.375$ *and observe that* $\rho_1 < 0.2$. *Which gives* $x_2 = x_1 = [1.1978, 1.4346]^T$ *at this situation* x_1 *and* x_2 *are the same but the value of* Δ *is different so go next iteration*

Iteration 3

Start with $x_2 = [1.1978, 1.4346]^T$ *and* $\Delta_2 = 0.375$ $sg = [-0.0003797, 0.0000208]^T$, $\|sg\| = 0.00003802$, $sn = [-0.1930, -0.4622]^T$, $\|sn\| = 0.5009$

now compare the value of $\|sg\|$ *and* $\|sn\|$ *to* Δ_2 *then observe that* $\|sg\| < \Delta_4$ $\|s_n\| > \Delta_4 \Rightarrow$ *first find* α *by using the formula states on the above algorithm. then we get* s_k *and the ratio* ρ_k*=-0.1781 this shows that the actual and predicted reduction is not related and so the new point is not accepted we need to reduced the trust region radius and construct a new predicted model therefore, the starting point is* $x_3 = x_2 = (1.1978, 1.4346)$ *and the trust region is reduced to* $\Delta_3 = 0.25\Delta_2 = 0.09375$

Iteration 4

Start with $x_3 = (1.1978, 1.4346)$ $\Delta_3 = 0.09375$, $sg = [-0.0003797, 0.0000208]^T$, $\|sg\| = 0.00003802$, $sn = [-0.1930, -0.4622]^T$, $\|sn\| = 0.5009$

now compare the value of $\|sg\|$ *and* $\|sn\|$ *to* Δ_3 *then observe that then observe that* $\|sg\| < \Delta_3$ *and* $\|sn\| > \Delta_3$ *we get* $s_3 = [-0.0936, 0.0051]^T$ *and* $\|p_3\| = 0.0937$

The ratio $\rho_3 = 1.0003$.*It follows* $\rho_3 > 3/4$ *and* $\|p_3\| \neq \Delta_3$ *therefore* $\Delta_4 = \Delta_3$ *and* $\rho_3 > 0.2$ *which gives* $x_4 = x_3 + p_3$=$[1.1614, 1.3482]^T$

The following table is a ***summery*** *for the improving process*

k	norm(g)	x_1	x_2	r	obj
1	229.16928	1.19775	1.43461	1.50000	19.40000
2	0.39793	1.19775	1.43461	0.37500	0.03911
3	0.39793	1.19775	1.43461	0.09375	0.03911
4	0.39793	1.16140	1.34819	0.09375	0.03911
5	0.63759	1.12392	1.26226	0.09375	0.02609
6	0.69343	1.08535	1.17681	0.09375	0.01544
7	0.72179	1.04562	1.09189	0.18750	0.00742
8	0.74827	1.01017	1.01919	0.18750	0.00229
9	0.58501	1.00204	1.00402	0.18750	0.00026
10	0.03329	1.00003	1.00005	0.18750	0.00000
11	0.00187	1.00000	1.00000	0.18750	0.00000
12	0.00000	1.00000	1.00000	0.18750	0.00000

Table 3.1: Results of single dogleg algorithm

The minimizer is [1.0000,1.0000] and the min value is: 5.1845e-016.

Note that: After 12 iteration we get the minimum result $f(x_1^*, x_2^*)\ is\ \ near\ \ \ 0$ at $(x_1^*, x_2^*) = (1, 1)$.
Again by considering the same initial point and condition we look at the optimization results (see Table 3.2) when double dogleg algorithm is used.

The minimizer is: $x = [1.0000; 1.0000]^T$, the min value: is 1.8961e-015. After only 11 iteration we get the minimum result $f(x_1^*, x_2^*)$ is near to zero at $(x_1^*, x_2^*) = (1, 1)$. Comparing results from the table(3.1) and (3.2) the double dogleg method is faster convergence than dogleg method. At the starting points [1.2,1] which is closed to the minimum point of the function .
Again by considering same condition for both Newton and steepest descent method see table (3.3) and (3.4) from table(3.3) the Newton method takes

k	norm(g)	x_1	x_2	obj
1	1506.36698	1.19775	1.43461	19.40000
2	1349.39472	1.00020	0.96137	0.03911
3	1014.45820	1.00018	1.00035	0.15231
4	1001.88376	1.00000	1.00000	0.00000
5	1001.60065	1.00000	1.00000	0.00000
6	1001.60064	1.00000	1.00000	0.00000

Table 3.2: Results of Newton method algorithm

to converge only 6 iterations.

k	norm(g)	x_1	x_2	obj
1	12	1	1	0.9836052598082
2	23	1	1	0.0013541700525
3	34	1	1	0.0010591698436
4	43	1	1	0.0010581875594
5	53	1	1	0.0010569161569
6	64	1	1	0.0010534444273
7	67	1	1	0.0010283313350
8	78	1	1	0.0008623496692
9	87	1	1	0.0008606904813
10	97	1	1	0.0008595125969
11	107	1	1	0.0008583464013
12	117	1	1	0.0008571934154
13	127	1	1	0.0008560540807
14	137	1	1	0.0008549303048
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
8252	82175	1	1	0.0000000000665
8253	82185	1	1	0.0000000000664
8254	82195	1	1	0.0000000000663

Table 3.3: Results of steepest descent method algorithm

From table (3.4) the steepest descent method takes to converge 8254 iterations. To the minimizer: $[1.0000; 1.0000]^T$ and the objective value 6.6284e-011.
search 2 we are taking the initial point which is far from the global minimum of the function (1,1). Take any point let say [100;100] the single dogleg algorithm converges after 103 iteration but the double dogleg algorithm converges after 100 iteration, so this shows that the double dogleg method is faster convergence than the dogleg methods for both cases. And the Newton method takes 5 iteration in this function Newton method is most fastest than single, double and the steepest descent method for any point, because the Hessian matrix of the given function is positive definite for each iteration.
Generally double dogleg method is faster convergence rate than the single dogleg by considering the same initial point and conditions(tolerance).

Example 2. *Consider the function*

$$f = -10x_1^2 + 10x_2^2 + 4sin(x_1.x_2) - 2x_1 + x_1^4. \tag{3.2}$$

Now to find an approximation solution of this function by using single dogleg, double dogleg method and Newton method?

Search 1: *The first search begins by choosing a start point very close to the minimum. Let take the point [2,-1], in this search the Newton method is more efficient since it reaches the local minimum in only 5 iterations whilst the Trust-region single Dogleg and double dogleg takes 6 iterations. This is to be expected since the major advantage of Newton's method is fast convergence when near the minimum, whilst the Trust region method usually begins by taking steps the cauchy point.*

Search 2: *In the second search the start the start point is moved only slightly further away from the minimum. let take [3,-2] but even this is enough to cause Newton's method to diverge. The Trust region (dogleg) performance as predicted by the theory converges and reaches one of the minimum in 23 iteration. The reason Newton's method failed is in most likely due to the fact the hessian was not positive definite at a particular iteration, this will certainly have caused divergence.*

Search 3: *It is not surprising the Newton method again fails when the starting so far from the minimum in the search, however what is even more surprising is the fast convergence of the trust region method the single dogleg needed 892 iteration and double dogleg method need 810 iteration to converge. when starting at [500,-560] which is a long distance away from the local minimum [2,-0.3].*

Conclusion

This report has explored the underlying theory(philosophy) of the trust region algorithm and its operations. The convergence properties of the algorithm when taking steps to the cauchy point has been examined and it has been shown that the double dogleg method is effective and robust in terms of speed or convergence rate than single dogleg method we observed in table (3.1 and 3.2). Newton method is fastest than the dogleg method and steepest method we observed table (3.1 ,3.2, 3.3 and 3.4) and search 1 of Example 2 but this method diverges when the initial point is far from the optimal point and hessian matrix is not positive definite at each iteration we observe in search 2 and 3 of Example 2. So the dogleg method mitigates the drawbacks of Newton method by using the cauchy point when the hessian is not positive definite. Due to this the dogleg method is an effective strategy since both the Cauchy point (guarantying the global convergence) and the full Newton step (ensuring the possibility to have faster rate of convergence) are incorporated into the possible step. In summary the trust region (dogleg) method is a modification of Newton's method with the aim of safe-guarding Newton's method from diverging by restricting the step size within the bounds of the trust region.

Annex

```
For Eg_1 use the following for all methods
*******************************************************
function T= fs(x)
T=100*(x(2)-x(1)^2)^2+(1-x(1))^2;
********************************************************
function T=gs(x)
T(1)=-400*x(1)*(x(2)-x(1)^2)-2*(1-x(1));
T(2)=200*(x(2)-x(1)^2);
T=T';
******************************************************
function T=hs(x)
T=[-400*x(2)+1200*x(1)^2+2, -400*x(1);-400*x(1), 200];
*******************************************************
*******************************************************

                       disp('ABAYE YESHITLA')
                        disp('Newton method')

x=input('Enter the column vector x');


    n=2;

  obj=fs(x)
  g=gs(x);
  H=hs(x);
  k=1;
  tol=0.00001;

disp('_____________________________________________')
disp('k       norm(H)     x1      x2         obj')
disp('_____________________________________________')
% fprintf('%3 %5.5f           %5.5f\n',k,norm(H),obj)

  while norm(g)>tol
      obj=fs(x);
      g=gs(x);
      H=hs(x);
      X=-inv(H)*g;
```

```
      x=x+X;
      k=k+1;
fprintf('%3.0f %5.5f   %5.5f   %5.5f  %5.5f\n',k-1,norm(H),x(1),x(2),obj)
  end
 disp('The minimizer is:'), x
 disp('The min value: is'), obj
 ************************************************************************
 ************************************************************************
 function steep


              disp('Steepest Descent method')


x=input('enter  x');


   n=length(x);
  a= 0.1;
  b = 0.5;
  obj=fs(x)
  g=gs(x);
  k=0;
  nf=1;


disp('______________________________________________')
disp('k            nf       x1     x2       obj')
disp('______________________________________________')
  while  norm(g) > 10^(-5)
    d = -g;
    t = 1;
    newobj = fs(x + t*d);
    nf = nf+1;
    while (newobj-obj)/t > a*g'*d
      t = t*b;
      newobj = fs(x + t*d);
      nf = nf+1;
    end
    x = x + t*d;
    obj=newobj;
```

```
    g=gs(x);
    k = k + 1;

  fprintf('%5.0f    %5.0f  %5.0f   %5.0f  %12.13f \n',k,nf,x(1),x(2),obj);
  end
x
fs(x)
*************************************************************************
*************************************************************************

                          disp('Single dogleg method')

 x=input('Enter the column vector x');


     n=2;


   obj=fs(x);
  g=gs(x);
  H=hs(x);
  k=1;
  tol=0.000001;
   r=1.5;
   q=0.2;
   eig(H);
disp('__________________________________________________________')
disp('k        norm(g)          x1          x2      r           obj')
disp('____________________________________________________________')
% fprintf('%3.0f      %5.5f      %5.4f    %5.5f\n',k,norm(g),r,obj)

   while norm(g)>tol
      obj=fs(x);
      g=gs(x);
      H=hs(x);
      gg=det(H);
   if g'*H*g<=0
tau=1;
      else
  tau=min(norm(g)^3/(r*g'*H*g),1);
      end
```

```
      sg=-tau*r*g/norm(g);
  if (eig(H))<=0
      sg=-tau*r*g/norm(g);
      sk=sg
      else
      sn=-inv(H)*g ;

   norm(sn);
  end
 if norm(sg)>=r
      sk=-r*g/norm(g);
 elseif norm(sn)<=r
        sk=sn;

   else

   b=sg'*(sg-sn);
  c= norm(sg-sn)^2;
  d=r^2-norm(sg)^2;
 p1=(b+sqrt(b^2+c*d))/c;
  p2=(b-sqrt(b^2+c*d))/c;
  if p1 >=0
p= p1;
elseif p2>=0
    p=p2;
     else
          break
end
  sk=sg+p*(sn-sg);
 end
  sk;

   m=fs(x)+g'*sk+0.5*sk'*H*sk;

  R=(fs(x)- fs(x+sk))/(-g'*sk-0.5*sk'*H*sk);
  norm(sk);
  if R<0.25
     r=0.25*r;
 elseif R>=0.75 & norm(sk)==r
      r=2*r;
       else
```

```
          r=r;
        end

        if R>q
            x=x+sk;

        else
            x=x;
        end

     k=k+1;
fprintf('%3.0f %5.5f%5.5f %5.5f  %5.5f %5.5f\n',k-1,norm(g),x(1),x(2),r,obj)
    end
 disp('The minimizer is:'), x
 disp('The min value: is'), obj
 ****************************************************************************
 ****************************************************************************

                          disp('Double dogleg method')

x=input('Enter the column vector x');
    n=2;

  obj=fs(x);
  g=gs(x);
  H=hs(x);
  k=1;
  tol=0.000001;
   r=1.5;
   q=0.2;
disp('_______________________________________________________')
disp('k      norm(g)x1             x2     r       obj')
disp('___________________________________________________________')
% fprintf('%3.0f  %5.5f   %5.4f    %5.5f\n',k,norm(g),r,obj)


  while norm(g)>tol
       obj=fs(x);
      g=gs(x);
      H=hs(x);
```

```
   if g'*H*g<=0
tau=1;
      else
  tau=min(norm(g)^3/(r*g'*H*g),1);
      end
      sg=-tau*r*g/norm(g);
      eig(H);
      if (eig(H))<=0

          sg=-tau*r*g/norm(g);
      sk=sg
      else
      sn=-inv(H)*g;
   norm(sg);
   norm(sn);
      end
   N=norm(g)^4/((g'*H*g)*(g'*inv(H)*g));
  T=0.8*N+0.2;

  if norm(sn)<=r
      sk=sn;
  elseif norm(sg)>=r
      sk=(-r*g)/norm(g);
 elseif norm(T*sn)<=r
      sk=r/norm(sn)*sn;
  else


      w=T*sn-sg;
      a=r^2-norm(sg)^2;
      b=sg'*w;
      alpha=a/(b+sqrt(b^2+norm(w)^2*a));
      sk=sg+alpha*w;
 end
    sk;
  m=fs(x)+g'*sk+0.5*sk'*H*sk;

   R=(fs(x)- fs(x+sk))/(-g'*sk-0.5*sk'*H*sk);

%   if norm(fsss(x)-m)==0
%       disp('no')
```

```
%       break
%   end
  norm(sk);
  if R<0.25
     r=0.25*r;
 elseif R>=0.75 && norm(sk)==r
      r=2*r;
       else
          r=r;
        end
          r;
        if R>q
            x=x+sk;
        else
            x=x;
        end




      k=k+1;
fprintf('%3.0f %5.5f %5.5f %5.5f %5.5f %5.5f\n',k-1,norm(g),x(1),x(2),r,obj)
  end,
 disp('The minimizer is:'), x
 disp('The min value: is'), obj
 ***************************************************************************
 ***************************************************************************
 For Eg_2 we use the following

************************************************************

function  r=fsss(x)

r=-10*x(1)^2+10*x(2)^2+4*sin(x(1)*x(2))-2*x(1)+x(1)^4;
************************************************************
    function r=gsss(x)
r(1)=-20*x(1)+4*cos(x(1)*x(2))*x(2)-2+4*x(1)^3;
r(2)=20*x(2)+4*cos(x(1)*x(2))*x(1);
*********************************************************
 function r= hsss(x)
r=[-20-4*sin(x(1)*x(2))*x(2)^2+12*x(1)^2,
```

```
-4*sin(x(1)*x(2))*x(1)*x(2)+4*cos(x(1)*x(2));
-4*sin(x(1)*x(2))*x(1)*x(2)+4*cos(x(1)*x(2)),20-4*sin(x(1)*x(2))*x(1)^2];
*****************************************************************
```

Bibliography

[1] Jorge Nocedal Stephen J.wright, Optimization, second edition, 2006.

[2] Wenyu sun, Ya xiang Yuan, optimization theory and methods Nonlinear programming, 2006.

[3] C.T.Kelly, Iteration methods for optimization, North carolina state university, 1999.

[4] Jorge Nocedal and Ya, xiang Yuan, Combining Trust region and Line search Techniques, 1998.

[5] J. Dennis Jr and H. Mei, two new unconstrained optimization algorithms which use function and gradient values, Journal of Optimization Theory and Applications, vol. 28, no. 4, 1979.

[6] J.E. Dennis (JR) and Schnabel R.B, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, Inc, Englewood Cliffs, New Jersy, 1983.

[7] V.Buljak, Inverse Analyses with model reduction, computational Fluid and soiled Mechanics.

[8] B.I.oruh, and O.M.Bamigbola, a new Dogleg method for solving the trust region sub problem, 2013.

[9] Gerald A.Shultz, RobertB.Schnabel, and Richard H.Byrd, Family of Trust region based algorithms for unconstrained optimization, 2005.

[10] Cheng jing wang, Dogleg paths and trust region methods with back tracking technique for unconstrained optimization, 2005.