GRADIENT-TYPE METHODS FOR UNCONSTRAINED OPTIMIZATION

By

GUAN HUI SHAN

A project report submitted in partial fulfilment of the
requirements for the award of Bachelor of Science (Hons.)
Applied Mathematics With Computing

Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

April 2016

DECLARATION OF ORIGINALITY

I hereby declare that this project report entitled "**GRADIENT-TYPE METHODS FOR UNCONSTRAINED OPTIMIZATION**" is my own work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : ____________________________

Name : ____________________________

ID No. : ____________________________

Date : ____________________________

APPROVAL FOR SUBMISSION

I certify that this project report entitled "**GRADIENT-TYPE METHODS FOR UNCONSTRAINED OPTIMIZATION**" was prepared by **GUAN HUI SHAN** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Hons.) Applied Mathematics With Computing at Universiti Tunku Abdul Rahman.

Approved by,

Signature : ___________________________

Supervisor : ___________________________

Date : ___________________________

ACKNOWLEDGEMENTS

I wish to express my earnest appreciation towards peoples who have guided me for the whole project. This project would not done perfectly without their dynamic guidance, help, participation and support.

Firstly, I wish to express my profound appreciation to my supervisor, Dr Mahboubeh Farid for her supervision and support throughout the journey. Her patience, enthusiasm and valuable contribution had made this project interesting. The efforts we had put together will surely become part of the valuable memory of my university life.

Next, I wish to express my sincere appreciation to my parents and my brother for their continuous encouragement and unfailing support throughout my studies. This achievement would be impossible without them. My appreciation to them is definitely more than simple words.

Lastly, I am also grateful to my friends who lend me their hand throughout my university life and guided me a lot regarding the project.

GUAN HUI SHAN

GRADIENT-TYPE METHODS FOR UNCONSTRAINED OPTIMIZATION

GUAN HUI SHAN

ABSTRACT

In this project, different gradient type methods which can be applied to solve an unconstrained optimization problem have been investigated. This project focuses on Barzilai and Borwein Gradient method, Monotone Gradient method via weak secant equation and Monotone Gradient method via Quasi-Cauchy Relation. Iterative scheme for each method was studied. To apply each of these methods, the functions was assumed to be convex and twice continuous differentiable. In yields of the application, a few standard unconstrained functions have been chosen for testing purposes. The results obtained show the number of iterations used for getting an optimal point. The result were used for analyzing the efficiency of the methods studied. Two comparisons had been made in this project. First is the Barzilai and Borwein Gradient method with Monotone Gradient method via weak secant equation and the second is Barzilai and Borwein Gradient method with Monotone Gradient method via Quasi-Cauchy Relation. These comparisons show that the Monotone Gradient type methods perform better as compared to the Barzilai and Borwein Gradient method. The number of iterations clearly was not affected by the dimension. Finally, verification for the two proposed algorithms was done to show the flow of the algorithms.

TABLE OF CONTENTS

List of Figures

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Everyone is expected to ba an investor in the future and none of us will go for investment with no return or with high risk. Conversely, everyone is looking for low risk investment and high return rate. If you are a manager of a company or a manufacturer of production sites, you will definitely aim for high profit and operate with the highest efficiency. Besides, you will definitely designate your employees into various departments and hope for the lowest overhead costs.

Under certain circumstances, one will need to make decisions. Goal(s) we wish to achieve is maximize the benefit with minimum effort. How can these physical decisions work towards a perfect outcome? Undoubtedly, this can be achieved via optimization. Optimization is a process to obtain the best result under any circumstances given. Through this process, we can find the condition that gives us the maximum or minimum value depending on the situation. It is expressed with a function containing variables. It is important that we first identify the objective of any given problem, which can be in terms of cost, time, profit or any quantity represented by a number. The model must be classified before solving any optimization problem. A few examples are indicated. It can be categorized into continuous optimization versus discrete optimization, constrained optimization versus unconstrained optimization, deterministic optimization versus stochastic optimization and etc. This classification process is important as the algorithms used for solving the problem are various according to its type.

In this project, the focus is the unconstrained optimization. Investigation on methods of unconstrained optimization is important for many reasons. If the model design does not have any active constraint, then the problem will need to involve unconstrained function minimization algorithm to determine the direction and distance traveled. Besides, a constrained optimization problem can be transformed into unconstrained optimization problems using multiplier methods and the penalty function. Last but not least, unconstrained minimization technique is suitable because it is widely used in linear and nonlinear problems.

1-1 Objective

In this project, the main concern is to investigate a few gradient-type methods for solving unconstrained optimization problem. There is believe to be significant improvement for each method. Secondly, iterative scheme of each method, which it is the mathematical procedure to generate set of improving approximate solution was studied. Lastly, MATLAB codes were run for the numerical results. By observing the results, efficiency of each method based on the function used was compared.

1-2 Scope

The focus of this project is the unconstrained optimization problem. The main method in this project is the Gradient method. A few Gradient methods, which are Barzilai and Borwein Gradient method, Monotone Gradient method via weak secant equation and Monotone Gradient method via Quasi-Cauchy Relation were investigated. The functions considered are convex and twice continuous differentiable.

1-3 Problem Statement

The main task for solving an unconstrained optimization question is to minimize the given objective function which relies upon real variables with no restriction on their value.

Let $w \in R^n$ be a real vector with $n \geq 1$ component and function $t : R^n \to R$, such that t is continuous, convex and twice continuous differentiable function. Then, the unconstrained optimization problem can be expressed as:

$$\min t(w), \quad w \in R^n \tag{1.1}$$

1-4 Research Methodology

As stated in the objective section, one of the purposes for this project is to study gradient methods used for solving unconstrained optimization problems. In Project 1, research was done for optimization. Then, a study on some gradient type methods such as steepest descent method, Newton method, Barzilai and Borwein Gradient method and Monotone Gradient methods via Weak Secant Relation was done.

For each of the methods, an iterative scheme was applied to find the solution. It is a mathematical procedure to get a set of improving approximate solutions. The process was repeated until a close gradient difference was obtained. This minimum point can be calculated manually. However, the process was proved tedious. Hence, the value was calculated with the help of MATLAB code. We had chosen MATLAB as it is more user friendly and it can compute results faster than other traditional programming languages like C or Java.

Since different kinds of gradient type methods are being investigated, the comparison was being done using standard functions to test the efficiency of the methods. From the results, we will conclude the most efficient method.

1-5 Methodology and Planning

In this project, the main concern is to investigate the gradient type methods used for solving unconstrained optimization. Each of the method has an unique iterative scheme where it is a process to generate the approximate solutions.

In chapter 2, basic definition and brief history of optimization was shown. A simple step by step guideline for solving an optimization problem was also provided. It is then further break down into few examples of optimization. Then, summary related to the research topic was shown.

In chapter 3, terminologies and definitions used in this project was defined. Optimality condition for unconstrained optimization problems was stated. Next, before going deep into each method, the general view of gradient type methods was shown. A basic scheme was stated here to show general idea on how the gradient method works. It is then followed by the gradient type methods studied in this project, namely Steepest Descent method, Newton method, Quasi Newton method(general case), Barzilai and Browein method, Monotone Gradient method via weak secant equation and Monotone Gradient method via quasi-Cauchy relation.

In chapter 4, the numerical result of this project was shown. The results obtained were based on the standard function as attached in Appendix. Two comparisons between the studied methods, which are Barzilai and Browein method with Monotone Gradient method via Weak Secant Equation and Barzilai and Browein method with Monotone Gradient method via Quasi-Cauchy Relation was done. The result obtained from the comparison was recorded. The recorded results were based on the number of iterations until a close difference solution was obtained. A paragraph of discussion is included to discuss the efficiency of a particular method.

In chapter 5, which is the last chapter of this project, a brief summary of the research findings and discussion was shown. Finally, it was concluded that the project objectives set were met.

This chapter was eneded with timeline of this project which can be seen through two Gantt Chat. During the first week of project 1 semester, a suitable supervisor, Dr Mahboubeh was found and the project title was set. Then, the study on gradient type methods began and stretched till the end of project 2. The proposal was prepared in early weeks of project 1 and was submitted on week 7. It was then followed-up by changing the proposal into an interim report. Concurrently, the study on MATLAB software and code was also being carried out. Finally, at the end of project 1, the interim report was completed and submitted on week 12 and project 1 ended with oral presentation on week 13. The milestone of project 1 was summarized in figure below:

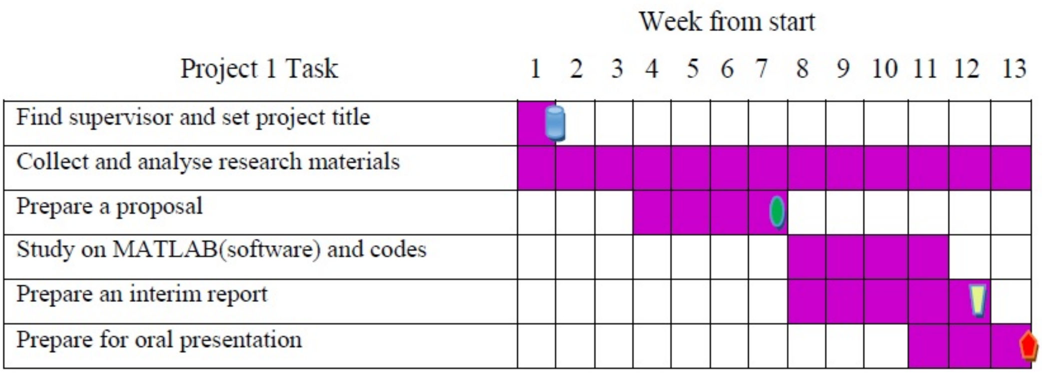

Figure 1.1: Milestone for Project 1

Found Dr Mahboubeh as supervisor and title for projects was set as "Gradient Type Methods for Unconstrained Optimization".

Proposal was submitted on 23.7.2015

Interim report was completed and was submitted on 27.8.2015

Project I was presented on 3.9.2015

Figure 1.2: Task Description for Project 1

Modification of interim report to final report began from week 1. The final report was type using LaTeX. The general case of Quasi Newton method and Monotone Gradient method via quasi-Cauchy relation was investigated in Project 2. Mid-semester monitoring form was submitted on week 7. For the subsequent week, the report was amended and finalized . Concurrently, similarity rate of the report was checked using Turnitin. Draft of complete report was submitted to supervisor on week 10. Finally, the finalized project report was submitted on week 12 and the project was ended by oral presentation on week 13. The milestone of project 2 is summarized in figure below:

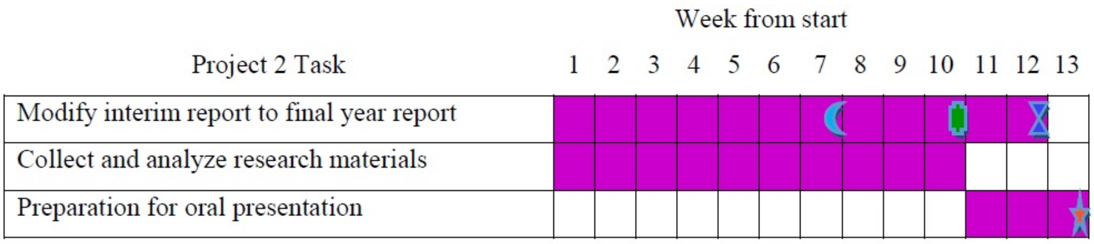

Figure 1.3: Milestone for Project 2

Mid-semester monitoring form was submitted on 2.3.2016

Draft of final report was submitted on 23.3.2016

Final report was submitted on 8.4.2016

Presentation for project II was presented on 14.4.2016

Figure 1.4: Task Description for Project 2

Chapter 2: Literature Review

The operations research optimization techniques started in England for investigating the tactics and strategy concerning troop operations amid the World War II. Management of England had fewer resources; hence they must use their resources in a more efficient way. In addition, a group of scientists played the role as advisors for the management and examined troops operation with a logical approach so that they could succeed in the war without battling. The growing of operational research optimization technique caused an industrial and economic boom. The management should think of some strategy to reach their objective. (Agarwal (2015)) So, what is an operations research optimization technique? Optimization may be characterized as investigation of deciding on the best solution to a given problem. This problem is usually a physical reality model. It includes the investigation of optimality criteria, determination of algorithmic strategies and investigation of structure of every strategy. (Fletcher (1987)) There are a few steps that need to be followed in order to solve an optimization problem. The initial step in solving optimization problem would be to construct a model. In this step, we will need to identify and express the objective of the problem as well as the variables used and the constraints of problem. The following step will be to determine the type of problem. There are many categories and hence it is important to classify it. This is an important step as it might affect the algorithm used for solving a problem. Lastly, we select suitable software to solve the problem. (Neos-guide.org (2016))

One of the popular methods used to solve unconstrained optimization problems is the steepest descent method. It is also referred to the gradient descent method and was designed by Cauchy in 1847. He claimed that gradient can be used for solving a non-linear equation problem. In addition, any function which is continuous should be decreasing in the early stage if the steps are taken along the negative gradient as the direction. He also specified that the computation for a convex quadratic function is an absolutely easy task. Hence, this method plays a vital role in developing the optimization theory. Lamentably, it performs slowly in real life problems and is also executed badly especially in poorly scaled condition problems. A problem is described as poorly scaled if the function value produced is having a large difference when we are changing the value of w in certain direction. For example, a function $t(w) = 10000000w_1^2 + w_2^2$. Clearly, this function is very sensitive towards w_1 but not w_2. (Nocedal & Wright (1999)) This disadvantage stands stronger when Xu (2008) stated that the process will be looping infinite times before we can find the minimum point which cause the long delay in finding solution for real life problem. Despite this, the method guarantees that the minimum solution will be found after looping many times if the particular minimum solution exists.

Steepest descent method lacks second derivative information, causing inaccuracy. Thus, a more effective method known as Newton method is proposed. Newton's work was done in year 1669 but it was published few years later. (Anstee (2006)) It uses second derivation, also known as the Hessian, bringing about better results. However, it is not easy to calculate Hessian manually especially when differentiation gets complicated. Even for some simpler differentiation, it may be time-consuming. (Bartholomew-Biggs (2005)) In addition, large storage space is needed to store the computed Hessian and thus it is computationally expensive. For example, if we have an Hessian with n dimension, we will need to have $O(n^2)$ storage space to store the Hessian. (Gibiansky (2014))

The famous method we used to solve an unconstrained problem is the Newton's method. However, for each iteration, we will need to compute the second derivation, which is the Hessian of a given function. This will need a lot of computing effort and in worst scenarios; the second derivation cannot be compute analytically. Therefore, in later year, Quasi Newton method was introduced. Instead of computing the Hessian, we use the approximation Hessian. There are a lot of updates can be used to calculate the Hessian approximation. The four famous update method which is normally been used are BFGS update, DFP update, PSB update and SR1 update. (Ding et al. (2010))

Later, the Barzilai and Borwein Gradient Method was proposed by Barzilai and Borwein in 1988. This method is preferable as compared to steepest descent method in terms of both computations and theories. This is because the steepest descent method performs badly in ill conditions. However, such difficulties would not occur if we are using the Barzilai and Borwein method. Additionally, it does not need to go through the line search process and it only needs very little computation. The higher efficiency and simplicity renders this method high attention from the masses. (Sun & Yuan (2006))

Hassan et al. (2009) was inspired by the Barzilai and Browein method which is superior to the steepest descent method. They endeavour to discover steplength formula that approximate inverse of Hessian using Quasi Cauchy equation which retain monotone property for every repetition. The method is known as the Monotone Gradient method via Quasi-Cauchy Relation. The approximation of inverse of Hessian is being stored in the diagonal matrix and hence it only required very little storage space, which is O(n). The repetition is also lesser as compared to the Barzilai and Browein method and it will converge to a desired point. Therefore, this method is a better choice for solving any problem.

Most recently, Leong et al. (2010) suggested a fixed step gradient type method to improve the Barzilai and Borwein method and it is known as the Monotone Gradient Method via Weak Secant Equation. In this method, accuracy of approximation had improved and information gathered in the previous iterations is used. The approximation is stored using diagonal matrix depending on the modified weak secant equation. The storage needed is also small for this method, which is just O(n). Therefore, it is clear that this method is better than BB method as it is easier for computation, accurate and does not require much storage.

CHAPTER 3: GRADIENT TYPE METHODS

Definitions of terminologies used in this project will be defined here. Then, optimality conditions for unconstrained optimization problem are stated. General view of gradient type methods will be shown for rough idea. Finally, the concerned Gradient type methods will be explored. The definitions and terminologies were referred from Fletcher (1987), Nocedal & Wright (1999), Sun & Yuan (2006)

3-1 Terminology and Definition

Definition 3.1. Comparing with the neighborhood N of point w^*, a point w^* is known as local minimizer if $t(w^*) \leq t(w)$ for all $w \in N$.

Definition 3.2. Comparing with all the w over R^n, a point w^* is known as global minimizer if $t(w^*) \leq t(w)$ for all $w \in R^n$.

Example 3.3. The figure below shows the local minimizer and global minimizer in a given graph.

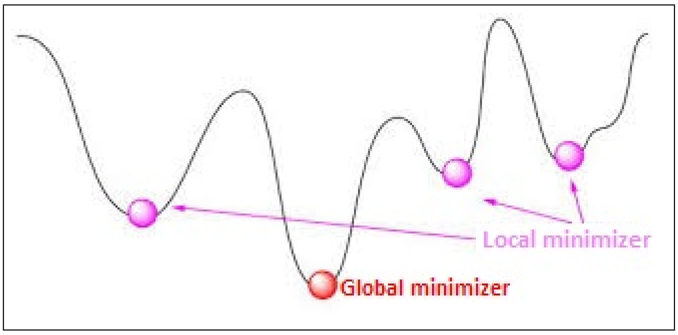

Figure 3.1: Local Minimizer and Global Minimizer

Definition 3.4. If a function $t : R^n \to R$ is continuous and $\frac{\partial t(w)}{\partial w_x}$ exist, $x = 1, 2, 3, ..., n$, then the function t is continuously differentiable at all $w \in R^n$. It is the gradient of function t at w and has notation

$$grad\, t = \nabla t = [\frac{\partial t(w)}{\partial w_1}, ..., \frac{\partial t(w)}{\partial w_n}]^T. \tag{3.1}$$

Definition 3.5. Suppose there is an open set $K \subset R^n$. Function t is said as continuously differentiable on K if it can be differentiate successively at all point on K. This function is expressed by $t \in C^1(K)$.

Definition 3.6. If a function $t : R^n \rightarrow R$ is continuous and $\frac{\partial^2 t(w)}{\partial w_g \partial w_h}$ exist, $g = 1, 2, 3, ..., n, h = 1, 2, 3, ..., n$, then the function t is twice continuous differentiable at all $w \in R^n$. It is the Hessian of function t and has notation

$$Hessian\, t = \nabla^2 t = \frac{\partial^2 t(w)}{\partial w_g \partial w_h}, \;\; 1 \leq g, h \leq n \tag{3.2}$$

Definition 3.7. Function t is said as twice continuously differentiable on K if it is able to differentiate twice continuously at all point of open set $K \subset R^n$. This function is expressed by $t \in C^2(K)$.

Definition 3.8. Assume that set $K \subset R^n$ and $w_1, w_2 \in K$. If $c_1 w_1 + (1 - c_1) w_2 \in K$, then K is a convex set.

Definition 3.9. Assume that set $K \subset R^n$ is a convex set which is nonempty and $t : K \subset R^n \rightarrow R,\ w_1, w_2 \in K\ and\ c_1 \in (0, 1)$. If $t(c_1 w_1 + (1 - c_1) w_2) \leq c_1 t(w_1) + (1 - c_1) t(w_2)$, then t is a convex function on K.

Definition 3.10. Fuction t is a concave function if $-t$ is a convex function.

Example 3.11. The figure below shows the example of concave and convex function.

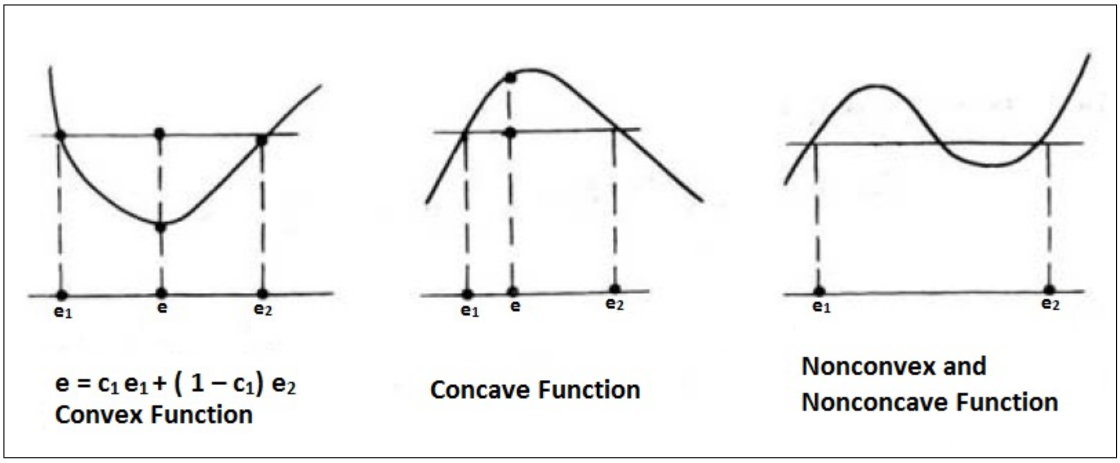

Figure 3.2: Concave Function and Convex Function

Theorem 3.12. *Below is some properties of a convex function:*

1. Suppose there is a convex set $K \subset R^n$ and a real number, $\gamma \geq 0$. If t is a convex function on K, then γt also a convex function on K.

2. Suppose there is a convex set $K \subset R^n$. If $t_1,\ t_2$ are the convex function on K, then $t_1\ +\ t_2$ is convex function on K.

3. Suppose there is a convex set $K \subset R^n$. If $t_1, t_2, ..., t_n$ are convex function on K and real numbers, $\gamma_1, \gamma_2, ..., \gamma_n \geq 0$, then $\sum_{x=1}^{n} \gamma_x k_x$ is a convex set on K.

Theorem 3.13. *Suppose $K \subset D$ is an open convex set and $t : D \subset R^n \rightarrow R$ is convex, then t is continuous on K. This is known as the first order characteristic of a given function.*

Theorem 3.14. *Suppose $K \subset R^n$ is an open convex set which is nonempty and function $t : K \subset R^n \rightarrow R$ is differentiable. If $t(m) \geq t(w) + \nabla t(w)^T(m - w)$, then function t is convex.*

Example 3.15. The figure below shows the example of First Order Characteristic for a convex function.

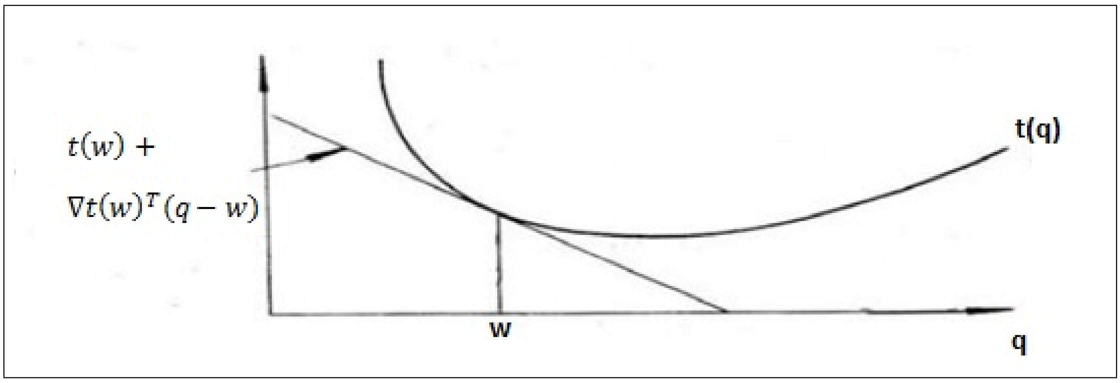

Figure 3.3: First Order Characteristic of Convex Function

Theorem 3.16. *Suppose there is a open convex set $K \subset R^n$ which is nonempty together with a continuous and twice differentiable function $t : K \subset R^n \rightarrow R$. For all points in K, if the second derivation is positive semidefinite, then the function t is convex.*

Theorem 3.17. *Suppose there is an open set K, a convex subset $P \subset K$ and a function $t : K \subset R^n \rightarrow R$. If the gradient of function t and ∇t is monotone, then t is convex on K.*

Theorem 3.18. *Suppose there is a nonempty open convex set* $K \subset R^n$ *and a twice continuous differentiable function* $t : K \subset R^n \to R$*. If the second derivation of* t*,* $\nabla^2 t(w)$ *is positive semidefinite, then* ∇t *is monotone on set* K*.*

3-2 Optimality Condition for Unconstrained Optimization

This project focuses on the unconstrained optimization problem with a continuous, twice differentiable and convex function t. The problem can be expressed with the equation below:

$$min\ \ t(w),\ w \in R^n \tag{3.3}$$

Now, optimality conditions including first order condition and second order condition are presented. The definition of local minimizer and global minimizer will be defined again.

Definition 3.19. Comparing with the neighborhood N of point w^*, a point w^* is known as local minimizer if $t(w^*) \leq t(w)$ for all $w \in N$.

Definition 3.20. Comparing with all the w over R^n, a point w^* is known as global minimizer if $t(w^*) \leq t(w)$ for all w.

Theorem 3.21. *Suppose that* $K \subset R^n$ *is a nonempty convex set and* $t : K \subset R^n \to R$*. If* t *is convex function, then a local minimizer point* $w^* \in K$ *will be also a global minimizer.*

In the following, the theorem of first order necessary condition (Theorem 3.22), second order necessary condition (Theorem 3.23) and second order sufficient condition (Theorem 3.24) are presented to make the definition of local minimizer and global minimizer clear.

Theorem 3.22. *Suppose there is an open set* K*,* $t : K \subset R^n \to R$ *and the function* t *is differentiable on* K*. If* w^* *is local minimizer of (3.3), then* $\nabla t(w^*) = 0$

Theorem 3.23. *Suppose there is an open set* K*,* $t : K \subset R^n \to R$ *and the function* t *is twice differentiable on* K*. If* w^* *is local minimizer of (3.3), then* $\nabla t(w^*) = 0$ *and* $\nabla^2 t(w^*)$ *is positive semidefinite.*

Theorem 3.24. *Suppose there is an open set K, $t : K \subset R^n \to R$ and the function t is twice differentiable on K. If $\nabla t(w^*) = 0$ and $\nabla^2 t(w^*)$ is positive definite, then w^* will be the local minimizer of function t.*

Definition 3.25. Suppose there is a point $w^* \in R^n$ and a function t where t is differentiable. If $\nabla t(w^*) = 0$, then the point is known as a critical point or a stationary point.

From the above definition and theorem, it is clear that if a point w^* is local minimizer, then it is also a critical point. Yet, the converse might not be true. A critical point w^* can be either a local maximizer or local minimizer.

Assuming that the objective function declared is a convex function, then the following theorem will hold.

Theorem 3.26. *Suppose $t : R^n \to R$ is a convex function and it is differentiable. If $\nabla t(w^*) = 0$, then w^* is a global minimizer.*

The final definition in this section give an important concept. This definition is regarding the descent direction.

Definition 3.27. Suppose a function $t : R^n \to R$ is differentiable at $w \in R^n$. If there is a vector $\varphi \in R^n$ and $\langle \nabla t(w), \varphi \rangle < 0$, then φ is known as descent direction of t at w.

(For proving of the theorems, see (Sun & Yuan (2006)))

3-3 Overview of Gradient-type Methods

The optimization method is frequently begins with an initial guess. The method refines repeatedly to approach to the optimal point. For every problem, we begin with an starting point, w_0 and an iterative sequence $\{w_x\}$ will be generated by means of some iterative rule to obtain an optimal solution, w^*. This iteration will be stopped once it satisfied the stopping criteria set earlier. Let α be a prescribed tolerance, which it is a very small positive value, then our termination criteria can be represented by

$$\|\nabla t(w_x)\| \leq \alpha \tag{3.4}$$

The gradient of the function t goes to zero when (3.4) is satisfied. The iterative sequence $\{w_x\}$ will converge to a desired point.

Let w_x be the value of w at $x-th$ iteration, d_x be the $x-th$ direction, β_x be the $x-th$ stepsize, then, the next w value can be calculated using

$$w_{x+1} = w_x + \beta_x d_x \tag{3.5}$$

Equation (3.5) shows that by employing different stepsize β_x and different direction d_x, we will have different methods. If $t(w_{x+1}) < t(w_x)$ at each iteration, it implies that $\beta_x d_x < 0$. Then d_x is a descent direction and this kind of optimization methods are known as descent method. In the following, the general scheme of optimization method is present.

Algorithm 3.1. (General scheme)

Step 1. Pick an initial point $w_0 \in R^n$ and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop.

Step 3. Based on the iterative scheme, compute the descent direction, d_x

Step 4. Compute the stepsize, β_x that make the function to be decreased, that is $t(w_x + \beta_x d_x) \leq t(w_x)$.

Step 5. Compute the next point value using $w_{x+1} = w_x + \beta_x d_x$. Set $x = x + 1$ and go to step 2.

3-4 Steepest Descent Method

The steepest descent method is also known as the gradient descent method. It is the fundamental and simplest method that used for solving optimization problem consisting n variables. There are two important factors to be recognized before solving the problem, which are the search direction and the steplength. The steplength enable us to know how far we should move in the direction. A function t and an initial point, w_0 was given. The goal to achieve is to look for a direction to cause a maximum decrease for the function t. It is a direction of negative gradient vector at the current point.

Let the function t be a continuously differentiable function at point w_x and the gradient function $\nabla t(w_x) \neq 0$. According to Taylor expansion,

$$t(w) = t(w_x) + (w - w_x)^T \nabla t(w_x) + o(\|w - w_x\|) \tag{3.6}$$

Now, let $w - w_x = \beta d_x$. The direction d_x is known as descent direction if it satisfy $d_x^T \nabla t(w_x) \leq 0$. Fixing the stepsize β, we find that the function value decreases faster if the value of $d_x^T \nabla t(w_x)$ is smaller.

Applying the Cauchy-Schwartz inequality, $\mid d_x^T \nabla t(w_x) \mid \leq \|d_x\| \|\nabla t(w_x)\|$, $d_x^T \nabla t(w_x)$ will have the smallest value if $d_x = -\nabla t(w_x)$. It implies that the steepest descent direction is $-\nabla t(w_x)$. Therefore, the steepest descent method have the following scheme

$$w_{x+1} = w_x - \beta_x \nabla t(w_x) \tag{3.7}$$

Now, the iterative scheme for the first proposed method: steepest descent method is present:

Algorithm 3.2. (Steepest Descent method)

Step 1. Pick a starting point $w_0 \in R^n$ and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop. Else, we set $d_x = -\nabla t(w_x)$

Step 3. Compute the stepsize, β_x such that $t(w_x + \beta_x d_x) = min\ t(w_x + \beta_x d_x)$

Step 4. Compute the next point value using $w_{x+1} = w_x + \beta_x d_x$. Set $x = x + 1$ and go to step 2.

3-4-1 Convergence of the Steepest Descent Method

In theory, the steepest descent method is significant method in optimization field and its convergence theory is important. In the following, convergence theorem for steepest descent method will be discussed.

Theorem 3.28. *Let Q be a positive constant. Suppose there is a function t which is twice differentiable continuously and $\|\nabla^2 t(w_x)\| \leq Q$. Considering an initial point,w_0 , the sequence generated by steepest descent method will end after many iterations, or* $\lim_{x\to\infty} \nabla t(w_x) = 0$ *or* $\lim_{x\to\infty} t(w_x) = -\infty$

From this section, it is clear that solving a problem using steepest descent method is very easy. However, the process for performing the steepest descent method is very slow in many real world problem. In most cases, this method performance is fine in the beginning stage. However, as it approaches the critical point, the process delayed with a zigzagging phenomena. The phenomenon of zigzagging occur as both gradients are perpendicular on the continuous iteration. Thus, the direction will be perpendicular too. The zigzagging phenomena is shown in figure below.

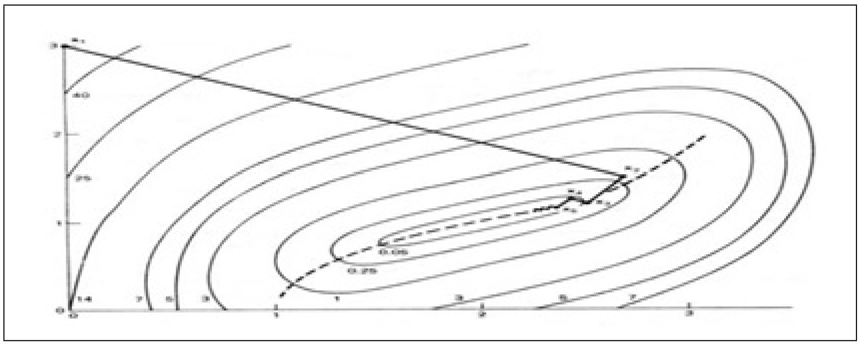

Figure 3.4: Zigzagging in Steepest Descend Method

(For proving of convergence of this method, see(Nocedal & Wright (1999)))

3-5 Newton Method

Suppose that $t : R^n \to R$ is continuous and twice differentiable function, there exist a point $w_x \in R^n$ and the Hessian $\nabla^2 t(w_x)$ is a positive definite matrix. Let $s = w - w_x$. To develop the model, the function can be expanded by the Taylor series

$$t(w_x + s) \approx q^{(x)}(s) = t(w_x) + \nabla t(w)^T s + \frac{1}{2} s^T \nabla^2 t(w_x) s \tag{3.8}$$

By minimizing the quadratic approximation, q^k,we will have the Newton formula:

$$w_{x+1} = w_x - [\nabla^2 t(w_x)]^{-1} \nabla t(w_x) \tag{3.9}$$

If we denote $G_x = \nabla^2 t(w_x)$ and $g_x = \nabla t(w_x)$, then we can rewrite the Newton formula as follows:

$$w_{x+1} = w_x - G_x^{-1} g_x \tag{3.10}$$

The Newton method's direction is $s_x = w_{x+1} - w_x = -G_x^{-1} g_x$. It yields that the direction will be a downhill direction as $-g_x^T G_x^{-1} g_x < 0$ if G_x is a positive definite matrix.

Now, we present the Newton method algorithm:

Algorithm 3.3. (Newton method)

Step 1. Pick an starting point $w_0 \in R^n$ and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop. Else, we compute $s_x = -G_x^{-1} g_x$

Step 3. Compute the next point value using $w_{x+1} = w_x + s_x$. Set $x = x + 1$ and go to step 2.

Newton's method is promising as it applies the Hessian matrix which carries more information about the curvature. Yet, the calculation of Hessian is very difficult in many situations. It needs a lot of computational effort and requires $O(n^2)$. In addition, the Hessian may not be available analytically.

3-5-1 Convergence of the Newton Method

Before showing the convergence theorem, we will first give the definition of Lipschitz continuous function.

Definition 3.29. Let $t : R^n \to R$ be our function, $K \subset D$ be our set and two points w_0 and w_1 is in K. If there exist a constant $Z > 0$ and $\|t(w_1) - t(w_0)\| \leq Z\|w_1 - w_0\|$, then the function t is Lipschitz continuous.

Now, we present the convergence theorem for Newton method. The function t we are using is assumed to be twice differentiable and the second derivation, the Hessian of our function is Lipschitz continuous satisfying sufficient condition, then the following theorem will hold.

Theorem 3.30. *If a function t can be differentiate twice and Hessian of the given function is Lipschitz continuous around neighborhood of optimal point, w^* satisfying the sufficient condition. Then,*

1. if the initial point w_0 is near to the optimal point w^*, then the array of iteration will be converging to the optimal point.

2. the convergence rate of the sequence $\{w_x\}$ is quadratic

3. the gradient norm sequence $\{\|\nabla t_x\|\}$ will be converging quadratically to 0.

(For proving of convergence and theorems for this method, see (Anstee (2006)) and (Nocedal & Wright (1999)))

3-6 Quasi Newton Method

One of the popular technique to solve unconstrained problem is through Newton method which we discussed in the section earlier. In Newton method, we need to compute the second derivative of the function, which is also known as the Hessian. However, it needs a lot of computational effort and it may be unavailable in some case. Hence, rather than calculating the Hessian matrix, we might think of creating a Hessian approximation. For instance, B_x in Quasi Newton method. We expect the sequence $\{B_x\}$ has positive definiteness, the direction $d_x = -B_x^{-1} g_x$ heading down and it acts similar to Newton method. There are four well known updates used for updating the Hessian approximation in Quasi Newton methods, which are BFGS updates which is introduced by Broyden, Fletcher, Goldfarb and Shanno in 1970 (see Hussain & Suhhiem (2015), FinMath (2013)), DFP updates which developed by Davidon, Fletcher and Powellsee (see Holmstr (1997)), PSB updates which is suggested by Powell and SR1 updates (see Ding et al. (2010)).

We do not investigate deep into each updates, however, we will give the general algorithm for Quasi Newton method here,

Algorithm 3.4. (Quasi Newton method)

Step 1. Pick an initial point $w_0 \in R^n$, initial inverse Hessian approximation, B_0 and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop. Else, we compute $d_x = -B_x g_x$

Step 3. Compute the steplength β_x

Step 4. Compute the next inverse Hessian approximation using the formula of a particular updates

Step 5. Compute the next point value using $w_{x+1} = w_x + \beta_x d_x$. Set $x = x + 1$ and go to step 2.

3-7 Barzilai and Borwein (BB) Gradient Method

As we discussed earlier, the weakness of steepest descent method is its poor performance especially in ill condition problem. Hence, Barzilai and Borwein offered another gradient method with modified steplength in 1988. The BB is in the frame of

Newton method, but we do not need to compute the Hessian. In their approach, the steplength is derived drom a two point approximation via secant equation. We can express the iterative scheme for BB method as follow:

$$w_{x+1} = w_x - D_x^{-1} g_x \tag{3.11}$$

where $D_x = (\frac{1}{\theta_x})I$. Matrix D_x should satisfy in the secant equation, therefore we compute θ_x as follows

$$min\|D_x s_{x-1} - y_{x-1}\|_2 \tag{3.12}$$

Let us denote $s_{x-1} = w_x - w_{x-1},\ y_{x-1} = g_x - g_{x-1}$. Then, we get the equation for computing BB stepsize as follows:

$$\theta_x = \frac{s_{x-1}^T s_{x-1}}{s_{x-1}^T y_{x-1}} \tag{3.13}$$

Alternatively, by minimizing $min\|s_{x-1} - D_x^{-1} y_{x-1}\|_2$ with respect to θ_x, we can rewrite formula for calculating α as

$$\theta_x = \frac{y_{x-1}^T s_{x-1}}{y_{x-1}^T y_{x-1}} \tag{3.14}$$

Now, we present the BB Method algorithm.

Algorithm 3.5. (BB method)

Step 1. Pick an initial point $w_0 \in R^n$, an initial stepsize, θ_0 and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop. Else, we suppose that $s_x = -\theta_x g_x$.

Step 3. Compute the next point value using $w_{x+1} = w_x + s_x$

Step 4. Compute the value y_x by $y_x = g_{x+1} - g_x$

Step 5. Compute the BB steplength using (3.13) or (3.14)

Step 6. Set $x = x + 1$ and go to step 2.

3-7-1 Convergence of the Barzilai and Borwein (BB) Gradient Method

Now, we present the convergence theorem for Barzilai and Borwein method. The function t we are using is assume to be a convex quadratic function. Then, a sequence of $\{w_x\}$ which is yield using the algorithm will converge to a point, w^*. The rate of convergence will be R-superlinear.

(For proving of convergence for this method, see (Barzilai & Borwein (1988)))

3-8 Monotone Gradient Method via Weak Secant Equation

Assuming that we have a matrix Q which where it is positive definite and also symmetric. Consider the following convex quadratic function:

$$t(w) = \frac{1}{2}w^T Qw - b^T w \tag{3.15}$$

Now, suppose that we generated a sequence $\{w_x\}$ by steepest descent method with initial point w_0. As the gradient of function $g_x = \nabla t(w_x) = Qw_x - b$, we have

$$g_{x+1} = (I - \theta_x Q)g_x \tag{3.16}$$

We suppose that matrix Q has different eigenvalues, $0 < \eta_1 < \eta_2 < ... < \eta_n$, together with $g_1^{(p)} \neq 0$ for all $p = 1, 2, ..., n$. We know that under orthogonal transformation, the gradient method will be remain unchanged. Additionally, gradient components can be merged if the gradient components are corresponding to the same eigenvalues. Hence, We can suppose that matrix Q is

$$Q = diag(\eta_1, \eta_2, ..., \eta_n) \tag{3.17}$$

Let $g_x = (g_x^{(1)}, g_x^{(2)}, ..., g_x^{(n)})^T$ and by (3.16) and (3.17), we will have

$$g_{x+1}^{(p)} = (1 - \theta_x \eta_p)g_x^{(p)} \tag{3.18}$$

We may calculate the stepsize, θ_x using the following formula:

$$\theta_x = \frac{2}{\eta_1 + \eta_n} \tag{3.19}$$

However, the value of η_1 and η_n is normally an unknown value. So, the calculation of stepsize by (3.19) is only good in theory. Hence, another good gradient method named Monotone gradient method was introduced. Let $w_x = (w_x^1, w_x^2, ..., w_x^n)^T$ and θ_x as stepsize in negative gradient direction, then we have the following updating scheme

$$w_{x+1} = w_x - \theta_x g_x \tag{3.20}$$

The equation (3.20) can be also written as $w_{x+1} = w_x - L_x^{-1} g_x$, where $L_x = diag(\frac{1}{\theta_x^{(1)}}, \frac{1}{\theta_x^{(2)}}, ..., \frac{1}{\theta_x^{(n)}})$. Since the matrix L_x is a diagonal matrix, it only needs the storage of $O(n)$. Let $s_x = w_{x+1} - w_x$, $y_x = g_{x+1} - g_x$ and $A = diag((s_x^{(1)})^2, (s_x^{(2)})^2, ..., (s_x^{(n)})^2)$.

The value of θ_{x+1}, which is the diagonal element of matrix L_{x+1} can be updated using the formula as follow:

$$\theta_{x+1} = \theta_x + \frac{(s_x^T y_x - s_x^T L_x s_x)(s_x^{(p)})^2}{tr(A^2)},\ p = 1, 2, ..., n \tag{3.21}$$

If the updated diagonal matrix does not preserve the positive definiteness property, then it will be replaced by the previous diagonal updating matrix.

Now, we present the Monotone Gradient method algorithm via Weak Secant Equation.

Algorithm 3.6. (Monotone Gradient method via Weak Secant Equation)

Step 1. Pick an initial point $w_0 \in R^n$, an initial positive definite diagonal matrix, $L_0 = I$ and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop.

Step 3. Compute $L_x^{-1} = diag(\frac{1}{\theta_x^{(1)}}, \frac{1}{\theta_x^{(2)}}, ..., \frac{1}{\theta_x^{(n)}})$,
where $\theta_{x+1}^{(p)} = \theta_x^{(p)} + \frac{(s_x^T y_x - s_x^T L_x s_x)((s_x^{(p)})^2)}{tr(A^2)},\ p = 1, 2, ..., n$

Step 4. If $L_x > 0$ is violated, then we put $L_x^{-1} = L_{x-1}^{-1}$. Else, we keep the L_x^{-1} which we computed in Step 3.

Step 5. Compute the value of next point using $w_{x+1} = w_x - L_x^{-1} g_x$.

Step 6. Set $x = x + 1$ and go to step 2.

3-8-1 Convergence of the Monotone Gradient Method via Weak Secant Equation

We will also illustrate the convergence theorem for Monotone Gradient method via Weak Secant Equation. This convergence theorem is important whenever the method is being applied. The function t we are using is assumed to be bounded below and strictly convex.

Theorem 3.31. *Suppose that the sequence $\{w_x\}$ is yield using the algorithm for Monotone Gradient method via Weak Secant Equation and a point w^* is an unique minimizer for function t, then $g_x = \nabla t(w_x) = 0$ will be retained for $x \geq 1$ or $\lim_{x\to\infty} \|g_x\| = 0$ and the sequence $\{w_x\}$ will be converging R-linearly to a desired point w^**

(For proving of convergence for this method, see (Leong et al. (2010)))

3-9 Monotone Gradient method via Quasi-Cauchy Relation

Suppose that we have a matrix Q such that the matrix is positive definite and symmetric. Think over the below convex quadratic function:

$$t(w) = \frac{1}{2}w^T Q w - b^T w \tag{3.22}$$

Now, we utilized an initial point w_0 for generating a sequence $\{w_x\}$ by steepest descent method. Combining the iterative scheme of steepest descend method with the above mentioned quadratic function, we will have

$$g_{x+1} = g_x - Q\theta_x g_x \tag{3.23}$$

Matrix Q will definitely has distinct eigenvalues, $0 < \eta_1 < \eta_2 < ... < \eta_n$, together with $g_1^{(p)} \neq 0$ for all $p = 1, 2, ..., n$. Under orthogonal transformation, the gradient method will be remain unchanged. In addition, all the gradient components can be merged if the gradient components are referring to the same eigenvalues. Therefore, we can write matrix Q as

$$Q = diag(\eta_1, \eta_2, ..., \eta_n) \tag{3.24}$$

Combining all the g_x and we express it as $(g_x^{(1)}, g_x^{(2)}, ..., g_x^{(n)})^T$ and by (3.23) and (3.24), we will have

$$g_{x+1}^{(p)} = (1 - \theta_x \eta^{(p)}) g_x^{(p)} \tag{3.25}$$

From (3.25), we acquire some connection , where

$$|g_{x+1}^{(p)}| \leq |1 - \theta_x \eta^{(p)}||g_x^{(p)}| \tag{3.26}$$

Hence, we may calculate the stepsize, θ_x using the following formula:

$$\theta_x^{(p)} = \frac{1}{\eta^{(p)}}, \;\; p = 1, 2, ..., n \tag{3.27}$$

Using the value of stepsize, we can have our updating scheme as

$$w_{x+1}^{(p)} = w_x^{(p)} - \theta_x^{(p)} g_x^{(p)} \tag{3.28}$$

Let L_x be the diagonal matrix containing all the stepsizes, then we can rewrite (3.28) as

$$w_{x+1} = w_x - L_x g_x \tag{3.29}$$

Yet, the value of $\eta^{(p)}$ is often an unknown value. Hence, we could not compute the value using the formula and we can only say that the formula is good in theory. Thus, usage of approximation of Q^{-1} would be a better choice. This approximation should always fulfill the quasi-Newton equation requirement, at which $Q^{-1}y_x = s_x$, given the value of s_x is $s_x = w_{x+1} - w_x$ and the value of y_x is $y_x = g_{x+1} - g_x$. For us to approximate the value of Q^{-1} correctly by updating L_{x+1}, we must make sure that L_{x+1} fulfill quasi-Caichy equation. Since the matrix L_x is a diagonal matrix, it only needs the storage of $O(n)$. Let $s_x = w_{x+1} - w_x,\ \ y_x = g_{x+1} - g_x$ and $A = diag((y_x^{(1)})^2, (y_x^{(2)})^2, ..., (y_x^{(n)})^2)$. The value of l_{x+1}, which is the diagonal element of matrix L_{x+1} can be updated using the formula as follow:

$$l_{x+1} = l_x + \frac{(s_x^T y_x - y_x^T L_x y_x)(y_x)^2}{tr(A^2)},\ p = 1, 2, ..., n \tag{3.30}$$

We can not assure that this method which utilizing (3.29) will be monotone forever. Hence, it is vital for us to comprise monotone strategy into it. Suppose that l_x^m is the smallest diagonal component in L_x, l_x^M is the largest diagonal component in L_x, then we present the algorithm for Monotone gradient type method via Quasi-Cauchy Relation as follow.

Algorithm 3.7. (Monotone Gradient method via Quasi-Cauchy Relation)

Step 1. Pick a starting point $w_0 \in R^n$, a starting positive definite diagonal matrix, $L_0 = I$, initiating $x = 0$ and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop.

Step 3. When $x = 0$, calculate the value of $w_0 - \frac{g_0}{\|g_0\|}$ and set it as w_1, then proceed to step 5. For other x value, we let $w_{x+1} = w_x - L_x g_x$ and update the value of L_{x+1} where $L_{x+1} = diag(l_{x+1}^{(1)}, l_{x+1}^{(2)}, ..., l_{x+1}^{(n)})$. The value of $l_{x+1}^{(p)},\ p = 1, 2, ..., n$ can be calculated using (3.30).

Step 4. If $L_{x+1} > 0$ or $l_x^m - \frac{(l_{x+1}^m)^{-1}(l_x^M)^2}{2} > 0$ is violated, then we put $L_{x+1} = \theta_x I$, such that the value of θ_x is calculated by:

$$\theta(x) = \begin{cases} (\frac{y_x^T s_x}{y_x^T y_x}), & if\ \frac{y_x^T s_x}{y_x^T y_x} < 2(l_x^m)^3 \\ 2(l_x^m)^3, & if\ \frac{y_x^T s_x}{y_x^T y_x} \geq 2(l_x^m)^3 \end{cases}$$

Else, we keep the L_{x+1} which we computed in Step 3.

Step 5. Set $x = x + 1$ and go to step 2.

3-9-1 Convergence of the Monotone Gradient method via Quasi-Cauchy Relation

Theorem 3.32. *Assuming that there is a function t which is convex quadratic. Then, a sequence of $\{w_x\}$ which is yield using the algorithm will converge to a point, w^*. The rate of convergence will be R-linear.*

(For proving of convergence for this method, see (Hassan et al. (2009)))

CHAPTER 4: RESULTS AND DISCUSSIONS

4-1 Comparison of Gradient Type Methods

Finally, comparison for efficiency of the proposed gradient type method was made. The selected gradient methods were tested using the standard unconstrained optimization function as attach in Appendix A-1. These standard unconstrained function are chosen from Andrei (2008) and More et al. (1981). MATLAB code were run to get the results. The original code was written by Farid,M. It was used in two of the papers, (refer Leong et al. (2010) and Hassan et al. (2009)). For this project, the code was developed. The stopping criteria was set as $\| \nabla t(w_x) \| \leq 10^{-4}$ for all functions. To avoid the process iterate in finite number of times, execution limitations were set. The program was only allowed to be executed within 1000 iterations. If it fails to show result within the boundary limitation, we record it as '-'. Else, the number of iteration is recorded.

4-1-1 Barzilai and Borwein Gradient Method and Monotone Gradient Method via Weak Secant Equation

First, we will compare the Barzilai and Borwein Gradient Method (BB Method) and Monotone Gradient Method via Weak Secant Equation (MonoGrad Method). The data obtained is shown from Table 4.1 to 4.3.

Table 4.1: Numerical result of BB Method and MonoGrad method

Test Function (dimension)	Starting point, w_0	BB Method	MonoGrad Method
Diagonal_2 (10)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{10})$	-	53
Diagonal_2 (40)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{40})$	-	140
Diagonal_2 (50)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{50})$	-	164
Diagonal_2 (80)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{80})$	-	220
Diagonal_2 (10^2)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{100})$	-	248
Diagonal_2 ($5 * 10^2$)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{500})$	-	579
Diagonal_5 (10)	(2,2,...,2,2)	-	19

Table 4.2: Numerical result of BB Method and MonoGrad method (continue)

Test Function (dimension)	Starting point, w_0	BB Method	MonoGrad Method
Diagonal_5 (50)	(2,2,...,2,2)	-	21
Diagonal_5 (10^2)	(2,2,...,2,2)	-	23
Diagonal_5 ($5 * 10^3$)	(2,2,...,2,2)	-	27
Diagonal_5 (10^4)	(2,2,...,2,2)	-	28
Diagonal_5 (10^5)	(2,2,...,2)	-	30
EG2 (20)	(1,1,...,1,1)	67	18
EG2 (50)	(1,1,...,1,1)	209	23
EG2 (10^2)	(1,1,...,1,1)	57	36
EG2 ($5 * 10^2$)	(1,1,...,1,1)	65	53
EG2 (10^3)	(1,1,..,1,1)	-	329
EG2 ($5 * 10^3$)	(1,1,...,1,1)	429	112
EG2 (10^4)	(1,1,...,1,1)	470	59
Extended Tridiagonal2 ($5 * 10^2$)	$(\frac{6}{10}, \frac{6}{10}, ..., \frac{6}{10})$	11	14
Extended Tridiagonal2 ($8 * 10^2$)	$(\frac{6}{10}, \frac{6}{10}, ..., \frac{6}{10})$	11	11
Extended Tridiagonal2 ($9 * 10^2$)	$(\frac{6}{10}, \frac{6}{10}, ..., \frac{6}{10})$	11	11
Extended Tridiagonal2 (10^3)	$(\frac{6}{10}, \frac{6}{10}, ..., \frac{6}{10})$	11	11
Extended Tridiagonal2 (10^4)	$(\frac{6}{10}, \frac{6}{10}, ..., \frac{6}{10})$	11	11
Extended 3 Exponential Terms (10)	$(\frac{-1}{10}, \frac{-1}{10}, ..., \frac{-1}{10})$	13	13
Extended 3 Exponential Terms (30)	$(\frac{-1}{10}, \frac{-1}{10}, ..., \frac{-1}{10})$	13	13
Extended 3 Exponential Terms (50)	$(\frac{-1}{10}, \frac{-1}{10}, ..., \frac{-1}{10})$	13	13
Extended 3 Exponential Terms (10^2)	$(\frac{-1}{10}, \frac{-1}{10}, ..., \frac{-1}{10})$	13	13

Table 4.3: Numerical result of BB Method and MonoGrad method (continue)

Test Function (dimension)	Starting point, w_0	BB Method	MonoGrad Method
Extended 3 Exponential Terms ($5*10^2$)	$(\frac{-1}{10}, \frac{-1}{10}, ..., \frac{-1}{10})$	13	13
Extended 3 Exponential Terms (10^3)	$(\frac{-1}{10}, \frac{-1}{10}, ..., \frac{-1}{10})$	13	13
Extended 3 Exponential Terms ($5*10^3$)	$(\frac{-1}{10}, \frac{-1}{10}, ..., \frac{-1}{10})$	13	13
Hager (10)	(2,2,...,2)	-	15
Hager (20)	(2,2,...,2)	23	20
Hager (40)	(2,2,...,2)	21	29
Hager (50)	(2,2,...,2)	24	35
Perturbed Quadratic (10)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	27	9
Perturbed Quadratic (30)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	39	12
Perturbed Quadratic (50)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	59	14
Perturbed Quadratic (90)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	79	23
Perturbed Quadratic (10^2)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	90	27
Perturbed Quadratic ($2*10^2$)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	98	32
Quadratic QF1 (10)	(1,1,...,1)	20	10
Quadratic QF1 (50)	(1,1,...,1)	39	15
Quadratic QF1 (10^2)	(1,1,...,1)	59	14
Quadratic QF1 ($5*10^2$)	(1,1,...,1)	190	30
Quadratic QF1 ($5*10^3$)	(1,1,...,1)	518	61
Quadratic QF2 (10)	(1,1,...,1)	6	6
Quadratic QF2 (50)	(1,1,...,1)	4	4
Quadratic QF2 (10^2)	(1,1,...,1)	4	4
Quadratic QF2 ($5*10^2$)	(1,1,...,1)	3	3
Quadratic QF2 (10^3)	(1,1,...,1)	3	3
Raydan_2 (10)	(1,1,...,1)	6	6
Raydan_2 (10^2)	(1,1,...,1)	6	4
Raydan_2 (10^3)	(1,1,...,1)	6	4
Raydan_2 (10^4)	(1,1,...,1)	6	4

Observing Table 4.1, Table 4.2 and Table 4.3, we find out that Monotone Gradient Method via Weak Secant Equation has higher efficiency as compared to Barzilai and Barzilai and Borwein Gradient Method. In few test function, Barzilai and Borwein method fails to converge to a stationary point within 1000 iterations, for example the Diagonal2 function and Diagonal5 function. Besides, from the result we obtain, we can judge that the dimension of a given function might not affect the number of iteration to be taken to reach a critical point. For example, in EG2 function, observing the number of iteration taken using the Barzilai and Borwein method, for the dimension = 20, which is quite small, it takes 67 iterations to reach a desired point. When the dimension is increased to almost twice of the previous example, where the dimension = 50, it takes more than 200 iteration to reach the optimum point. Observing the trend, we will expect that the number of iteration will be increased if the dimension number increased. However, this is not the case. When the dimension = 100, it only take 57 iterations to reach the critical point. Therefore, it is show that the number of iteration does not necessary depends on the dimension.

4-1-2 Barzilai and Borwein Gradient Method and Monotone Gradient method via Quasi-Cauchy Relation

Next, we compare the result between Barzilai and Browein Gradient method (BB method) and Monotone Gradient method via Quasi-Cauchy Relation (MonoCauchy method). The result is shown from Table 4.4 to 4.6.

Table 4.4: Numerical result of BB Method and MonoCauchy method

Test Function (dimension)	Starting point, w_0	BB Method	MonoCauchy Method
Broyden Tridiagonal(10^2)	(-1,-1,...,-1,-1)	43	39
Broyden Tridiagonal(10^3)	(-1,-1,...,-1,-1)	49	39
Broyden Tridiagonal($5 * 10^3$)	(-1,-1,...,-1,-1)	56	41
Broyden Tridiagonal(10^4)	(-1,-1,...,-1,-1)	58	42
Diagonal_2 (10)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{10})$	17	14
Diagonal_2 (70)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{70})$	50	44

Table 4.5: Numerical result of BB Method and MonoCauchy method (continue)

Test Function (dimension)	Starting point, w_0	BB Method	MonoCauchy Method
Diagonal_2 ($5 * 10^2$)	$(\frac{3}{1}, \frac{3}{2}, ..., \frac{3}{500})$	-	339
Diagonal_5 (10)	$(\frac{11}{10}, \frac{11}{10}, ..., \frac{11}{10})$	4	4
Diagonal_5 (10^2)	$(\frac{11}{10}, \frac{11}{10}, ..., \frac{11}{10})$	4	4
Diagonal_5 (10^3)	$(\frac{11}{10}, \frac{11}{10}, ..., \frac{11}{10})$	4	4
Diagonal_5 (10^4)	$(\frac{11}{10}, \frac{11}{10}, ..., \frac{11}{10})$	4	4
EG2(10)	(1,1,...,1,1)	50	19
EG2(10^3)	(1,1,...,1,1)	-	630
Extended Block Diagonal BD1(10)	$(\frac{1}{10}, \frac{1}{10}, ..., \frac{1}{10})$	14	11
Extended Block Diagonal BD1(10^2)	$(\frac{1}{10}, \frac{1}{10}, ..., \frac{1}{10})$	14	12
Extended Block Diagonal BD1(10^3)	$(\frac{1}{10}, \frac{1}{10}, ..., \frac{1}{10})$	14	13
Extended Block Diagonal BD1(10^4)	$(\frac{1}{10}, \frac{1}{10}, ..., \frac{1}{10})$	16	13
Extended Himmel-blau(10)	$(\frac{4}{10}, \frac{4}{10}, ..., \frac{4}{10})$	24	15
Extended Himmelblau(10^2)	$(\frac{4}{10}, \frac{4}{10}, ..., \frac{4}{10})$	24	16
Extended Himmelblau(10^3)	$(\frac{4}{10}, \frac{4}{10}, ..., \frac{4}{10})$	24	16
Extended Himmelblau(10^4)	$(\frac{4}{10}, \frac{4}{10}, ..., \frac{4}{10})$	24	17
Extended Trigonomer-tic(10)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	7	7
Extended Trigonomertic(10^2)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	46	41
Extended Trigonomertic(10^3)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	59	41

Table 4.6: Numerical result of BB Method and MonoCauchy method (continue)

Test Function (dimension)	Starting point, w_0	BB Method	MonoCauchy Method
Extended Trigonomertic($5 * 10^3$)	$(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2})$	62	49
Extended White & Hoist(10)	$(\frac{-12}{10}, \frac{-12}{10}, ..., \frac{-12}{10})$	-	150
Extended White & Hoist(10^2)	$(\frac{-12}{10}, \frac{-12}{10}, ..., \frac{-12}{10})$	-	169
Extended White & Hoist($5 * 10^2$)	$(\frac{-12}{10}, \frac{-12}{10}, ..., \frac{-12}{10})$	-	147
Extended PSC1(10)	$(3, \frac{1}{10}, ..., 3, \frac{1}{10})$	18	17
Extended PSC1(10^2)	$(3, \frac{1}{10}, ..., 3, \frac{1}{10})$	18	17
Extended PSC1(10^3)	$(3, \frac{1}{10}, ..., 3, \frac{1}{10})$	18	17
Extended PSC1(10^4)	$(3, \frac{1}{10}, ..., 3, \frac{1}{10})$	18	18
Raydan_1(30)	(1,1,...,1,1)	-	36
Raydan_1(10^2)	(1,1,...,1,1)	-	119
Raydan_1($5 * 10^2$)	(1,1,...,1,1)	-	630
Raydan_2(10)	(1,1,...,1,1)	6	6
Raydan_2(10^2)	(1,1,...,1,1)	6	6
Raydan_2(10^3)	(1,1,...,1,1)	6	6
Raydan_2(10^4)	(1,1,...,1,1)	6	6

Observing Table 4.4, Table 4.5 and Table 4.6, we find out that Monotone Gradient method via Quasi-Cauchy Relation has higher efficiency as compared to Barzilai and Barzilai and Borwein Gradient Method. In few test function, Barzilai and Borwein method fails to converge to a stationary point within 1000 iterations, for example the Generalized White and Holst function and Raydan1 function. Besides, from the result we obtain, we can judge that the dimension of a given function might not affect the number of iteration to be taken to reach a critical point. For example, in Extended Block Diagonal BD1 function, observing the number of iteration taken using the Barzilai and Borwein method, for the dimension = 10, which is quite small, it takes

14 iterations to reach a desired point. When the dimension is increased to 1000, which is very far larger than the previous dimension, it still able to converge to the optimum point within 14 iteration. Observing the trend, we will expect that the number of iteration will be increased if the dimension number increased. However, this is not the case. Looking on the Extended Himmelblau function, if we are trying to look for the optimal point using BB method, it takes the same number of steps to reach it. Therefore, it is show that the number of iteration does not necessary depends on the dimension.

4-2 Verification of Algorithm

In this section, we will verify the proposed method algorithm. The function we had chosen is the Raydan_2 function with dimension = 10 and the stopping criteria is set as $\| \nabla t(w_x) \| \leq 10^{-4}$. For each method, we give the algorithm again for references purposes. We will begin the verification for Barzilai and Browein(BB) method followed by the Monotone Gradient Method via Weak Secant Equation.

4-2-1 Barzilai and Browein method

First of all, we specify the algorithm for BB method.

Algorithm 4.1. (BB method)

Step 1. Pick an initial point $w_0 \in R^n$, an initial stepsize, θ_0 and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop. Else, we suppose that $s_x = \theta_x g_x$.

Step 3. Compute the next point value using $w_{x+1} = w_x + s_x$

Step 4. Compute the value y_x by $y_x = g_{x+1} - g_x$

Step 5. Compute the BB steplength using $\theta_x = \frac{s_{x-1}^T s_{x-1}}{s_{x-1}^T y_{x-1}}$ [refer (3.13)] or $\theta_x = \frac{y_{x-1}^T s_{x-1}}{y_{x-1}^T y_{x-1}}$ [refer (3.14)]

Step 6. Set $x = x + 1$ and go to step 2.

We will illustrate the iteration number, value of our test function and $\|\nabla t(w_x)\|$ in Table 4.7.

Table 4.7: Algorithm Verification for BB method

x	value of test function	$\|\nabla t(w_x)\|$
1	$1.7183 * 10^1$	1.6204
2	$1.0471 * 10^1$	$8.7419 * 10^{-1}$
3	$1.0101 * 10^1$	$4.7097 * 10^{-1}$
4	$1.0003 * 10^1$	$7.2105 * 10^{-2}$
5	$1.0000 * 10^1$	$4.9604 * 10^{-3}$
6	$1.0000 * 10^1$	$5.7488 * 10^{-5}$

In the first iteration, the value of $\|\nabla t(w_x)\|$ is 1.6204, which is clearly larger than the value of stopping criteria we had set earlier. Hence, we recompute the stepsize using the given formula and substitute into the updating formula as stated in step 3 in the algorithm. Then we increase the iteration number and repeat the same process. The value of $\|\nabla t(w_x)\|$ in step 2 is $8.7419 * 10^{-1}$, which is still larger than the predefined stopping criteria. Hence, the process in the algorithm is repeated again. Until the sixth iteration, the value of $\|\nabla t(w_x)\|$ is $5.7488 * 10^{-5}$, which is smaller than the value of the stopping criteria. The process is terminated and the optimum point is found. The value for the objective function we looking for is $1.0000 * 10^1$.

4-2-2 Monotone Gradient Method via Weak Secant Equation

Next, we give the verification for Monotone Gradient Method via Weak Secant Equation. We first specify the algorithm for the method.

Algorithm 4.2. (Monotone Gradient method via Weak Secant Equation)

Step 1. Pick an initial point $w_0 \in R^n$, an initial positive definite diagonal matrix, $L_0 = I$ and set a stopping tolerance $0 < \alpha < 1$

Step 2. Check if $\|\nabla t(w_x)\| \leq \alpha$. If yes, then stop.

Step 3. Compute $L_x^{-1} = diag(\frac{1}{\theta_x^{(1)}}, \frac{1}{\theta_x^{(2)}}, ..., \frac{1}{\theta_x^{(n)}})$,
where $\theta_{x+1}^{(p)} = \theta_x^{(p)} + \frac{(s_x^T y_x - s_x^T L_x s_x)((s_x^{(p)})^2)}{tr(A^2)}$, $p = 1, 2, ..., n$

Step 4. If $L_x > 0$ is violated, then we put $L_x^{-1} = L_{x-1}^{-1}$. Else, we keep the L_x^{-1} which we computed in Step 3.

Step 5. Compute the value of next point using $w_{x+1} = w_x - L_x^{-1} g_x$.

Step 6. Set $x = x + 1$ and go to step 2.

We will illustrate the iteration number, value of our test function and $\|\nabla t(w_x)\|$ in Table 4.8.

Table 4.8: Algorithm Verification for Monotone Gradient method via Weak Secant Equation

x	value of test function	$\|\nabla t(w_x)\|$
1	$1.7183 * 10^1$	3.1033
2	$1.298 * 10^1$	$8.1389 * 10^{-1}$
3	$1.040 * 10^1$	$2.8275 * 10^{-1}$
4	$1.0004 * 10^1$	$4.7399 * 10^{-2}$
5	$1.0000 * 10^1$	$2.2312 * 10^{-3}$
6	$1.0000 * 10^1$	$1.6564 * 10^{-5}$

In the first iteration, the value of $\|\nabla t(w_x)\|$ is 3.1033, which is clearly larger than the value of stopping criteria we had set earlier. Hence, we recompute the θ using the formula in step 3 and substitute into the diagonal matrix. We need to check whether the matrix is positive definite. If it is violated, we will take the matrix of the previous step. Else, we will use the matrix we had calculated. After confirming, we calculate the value of next point using formula stated in step 5. Then we increase the iteration number and repeat the same process. The value of $\|\nabla t(w_x)\|$ in step 2 is $8.1389 * 10^{-1}$, which is still larger than the predefined stopping criteria. Hence, the process in the algorithm is repeated again. Until the sixth iteration, the value of $\|\nabla t(w_x)\|$ is $1.6564 * 10^{-5}$, which is smaller than the value of the stopping criteria. The process is terminated and the optimum point is found. The value for the objective function we looking for is $1.0000 * 10^1$.

CHAPTER 5: CONCLUSION

Optimization was used since the second world war for military problem. The usage of optimization does not stop after the war but it continues to apply until now. In this era, all of us are using optimization in our daily life so that we can maximize the profit. There exist many ways for solving the optimization problem. In our project, we investigated the gradient type methods. For each of these methods, we apply iterative scheme for finding the optimal solution of a given problem. Besides, we also appreciate the algorithms which been introduced earlier as it was used to solve a given problem. We may not know when or how to start and end the iterative scheme if the algorithms do not appear. In our project, we make a few assumptions to the problem, which are our function is always convex and it can be differentiate twice successively.

We had run MATLAB code to check the number of iterations needed for the method to converge to a desired point, which is our optimal point. From the results obtained, we observed that the both monotone gradient type methods work better as compared to the BB method. We had restrict our iteration number to 1000 as any iterations above that are time consuming and were not efficient. From the result, we can conclude that BB method may fail to converge to the optimal point. Besides, we also realized that the iteration needed for converging to optimal point does not necessarily depends on the dimension.

REFERENCES

Agarwal, S., 2015. *Computer Based Optimization Techniques*, Alpha Science International Ltd.

Andrei, N., 2008. 'An unconstrained optimization test function condition', *Advanced Modeling and Optimization* **10**, 147–161.

Anstee, R., 2006. 'The newton-raphson method', Available at: `http://www.math.ubc.ca/~anstee/math104/newtonmethod.pdf`. [Accessed 3rd July 2015].

Bartholomew-Biggs, M., 2005. *Nonlinear Optimization with Financial Applications*, Boston: Kluwer.

Barzilai, J. & Borwein, J., 1988. 'Two-point step size gradient methods', *IMA Journal of Numerical Analysis* **8**, 141–148.

Ding, Y., Lushi, E. & Li, Q., 2010. 'Investigation of quasi-newton methods for unconstrained optimization', *International Journal of Computer Application* **29**, 48–58.

FinMath, 2013. 'Broyden-fletcher-goldfarb-shanno (bfgs) algorithm', Available at: `https://www.rtmath.net/help/html/9ba786fe-9b40-47fb-a64c-c3a492812581.htm`. [Accessed 11th March 2016].

Fletcher, R., 1987. *Practical Methods of Optimization*, Chichester : John Wiley and Sons.

Gibiansky, A., 2014. 'Hessian free optimization', Available at: `http://andrew.gibiansky.com/blog/machine-learning/hessian-free-optimization/`. [Accessed 19 August 2015].

Hassan, M. A., Leong, W. J. & Farid, M., 2009. 'A new gradient method via quasi-cauchy relation which guarantees descent', *Journal of Computational and Applied Mathematics* **230**, 300–305.

Holmstr, M., 1997. 'Davidon-fletcher-powell method', Available at: `http://user.it.uu.se/~matsh/opt/f9/node4.html`. [Accessed 11th March 2016].

Hussain, E. A. & Suhhiem, M. H., 2015. 'The convergence of bfgs method under strongly convex optimization problem', *International Research Journal* **3**(4), 90–103.

Leong, W. J., Hassan, M. A. & Farid, M., 2010. 'A monotone gradient method via weak secant equation for unconstrained optimization', *Taiwanese Journal of Mathematics* **14**, 413–423.

More, J., Garbow, B. & Hillstrom, K., 1981. 'Testing unconstrained optimization software', *ACM Transactions on Mathematical Software* **7**, 17–41.

Neos-guide.org, 2016. 'Introduction to optimization', Available at: `http://www.neos-guide.org/content/optimization-introduction`. [Accessed 26 February 2016].

Nocedal, J. & Wright, S., 1999. *Numerical optimization*, New York: Springer.

Sun, W. & Yuan, Y., 2006. *Optimization Theory and Methods*, Boston, MA: Springer Science+Business Media, LLC.

Xu, W., 2008. 'Method of steepest descent and its application', *IEEE Microwave and Wireless Components Letters* **12**, 24–26.

Appendix A: Test Function

Test function 1: Diagonal_2 Function

$$t(w) = \sum_{x=1}^{n} (exp(w_x) - \frac{w_x}{x}) \tag{A.1}$$

Test function 2: Perturbed Quadratic Function

$$t(w) = \sum_{x=1}^{n} x w_x^2 + \frac{1}{100} (\sum_{x=1}^{n} w_x)^2 \tag{A.2}$$

Test function 3: EG2 Function

$$t(w) = \sum_{x=1}^{n-1} sin(w_1 + w_x^2 - 1) + \frac{1}{2} sin(w_x^2) \tag{A.3}$$

Test function 4: Quadratic QF1 Function

$$t(w) = \frac{1}{2} \sum_{x=1}^{n} x w_x^2 - w_x \tag{A.4}$$

Test function 5: Quadratic QF2 Function

$$t(w) = \frac{1}{2} \sum_{x=1}^{\frac{1}{2}} x (w_x^2 - 1)^2 - w_x \tag{A.5}$$

Test function 6: Diagonal_5 Function

$$t(w) = \sum_{x=1}^{n} log(exp(w_x) + exp(-w_x)) \tag{A.6}$$

Test function 7: Extended Tridiagonal2 Function

$$t(w) = \sum_{x=1}^{n-1} (w_x w_{x+1} - 1)^2 + c(w_x + 1)(w_{x+1} + 1) \tag{A.7}$$

Test function 8: Hager Function

$$t(w) = \sum_{x=1}^{n} (exp(w_x - \sqrt{x} w_x) \tag{A.8}$$

Test function 9: Extended 3 Exponential Terms Function

$$t(w) = \sum_{x=1}^{\frac{n}{2}} (exp(w_{2x-1}+3w_{2x}-0.1)+wx[(w_{2x-1}-3w_{2x}-0.1)+exp(-w_{2x-1}-0.1)] \tag{A.9}$$

Test function 10: Broyden Tridiagonal Function

$$t(w) = (3w_1-2w_1^2)^2+\sum_{x=2}^{n-1}(3w_x-2w_x^2-w_{x-1}-2w_{x-1}+1)^2+(3w_n-2w_n^2-w_{n-1}+1)^2 \tag{A.10}$$

Test function 11: Extended Block Diagonal BD1 Function

$$t(w) = \sum_{x=1}^{\frac{n}{2}} (w_{2x-1}^2 + w_{2x} - 2)^2 + (exp(w_{2x-1} - 1) - w_{2x})^2 \tag{A.11}$$

Test function 12: Raydan_2 Function

$$t(w) = \sum_{x=1}^{n} (exp(w_x) - w_x) \tag{A.12}$$

Test function 13: Raydan_1 Function

$$t(w) = \sum_{x=1}^{n} \frac{x}{10}(exp(w_x) - w_x) \tag{A.13}$$

Test function 14: Extended Himmelblau Function

$$t(w) = \sum_{x=1}^{\frac{n}{2}} \frac{1}{2}(w_{2x-1}^2 + w_{2x} - 11)^2 + (w_{2x-1} + w_{2x}^2 - 7)^2 \tag{A.14}$$

Test function 15: Extended White and Holst Function

$$t(w) = \sum_{x=1}^{\frac{n}{2}} (1.5-w_{2x-1}(1-w_{2x}))^2+(2.25-w_{2x-1}(1-w_{2x}^2))^2+(2.625-w_{2x-1}(1-w_{2x}^3))^2 \tag{A.15}$$

Test function 16: Extended Trigonometric Function

$$t(w) = \sum_{x=1}^{n} ((n - \sum_{y=1}^{n} cos(w_y)) + x(1 - cos\ w_x) - sin\ w_x)^2 \tag{A.16}$$

Test function 17: Extended PSC1 Function

$$t(w) = \sum_{x=1}^{\frac{n}{2}} 100(w_{2x-1}^2 + w_{2x}^2 + w_{2x-1}w_{2x})^2 + sin^2(w_{2x-1}) + cos^2(w_{2x}) \tag{A.17}$$