People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

Setif 1 University Ferhat Abbas
Faculty of Sciences
Department of Mathematics

جامعـة سطيف 1 فرحـــات عباس
كليـــة العلـــــــوم
قســـــم الريـاضيـــــــات

Doctoral Thesis

submitted in fulfillment of the requirements for obtaining the degree of Doctor in Mathematics

Speciality: Nonlinear Optimization

Theme:

NEW SEARCH DIRECTIONS OF CONJUGATE GRADIENT METHODS FOR UNCONSTRAINED NONLINEAR OPTIMIZATION

Presented by

Mr. Mohamed Lamine OUAOUA

Supervisor: **Dr. Samia KHELLADI**
Co-supervisor: **Pr. Djamel BENTERKI**

Thesis defended on January 13th, 2026, in front of the jury composed of:

Mr.	Rachid ZITOUNI	Prof	Setif 1 University Ferhat Abbas	President
Mme.	Samia KHELLADI	MCA	Setif 1 University Ferhat Abbas	Supervisor
Mr.	Djamel BENTERKI	Prof	Setif 1 University Ferhat Abbas	Co- supervisor
Mme.	Rebiha ZEGHDANE	MCA	BBA University Mohamed El Bachir El Ibrahimi	Examiner
Mme.	Linda MENNICHE	MCA	Jijel University Mohamed Seddik Ben Yahia	Examiner
Mr.	Bachir MERIKHI	Prof	Setif 1 University Ferhat Abbas	Examiner
Mr.	Mohamed ACHACHE	Prof	Setif 1 University Ferhat Abbas	Guest

2026

ACKNOWLEDGMENTS

First of all, I would like to thank Allah (Exalted is He above all) who helped us fulfill this research work.

I extend my deep appreciation to my esteemed supervisor, Dr. Khelladi Samia, for her unwavering support, understanding, and invaluable contributions that have greatly enriched my experience. I also wish to express sincere thanks to my co-supervisor, Professor Benterki Djamel, for his guidance and assistance.

I am grateful to Professor Zitouni Rachid, for chairing the committee, as well as to Professor Merikhi Bachir, Dr. Zeghdane Rebiha and Dr. Menniche Linda for examining my thesis. My gratitude also to Professor Achache Mohamed for his kind acceptance to take part in the jury member invited.

Special thanks are due to my colleague and friend Hemici Youcef Elhamam for his unwavering encouragement, help and support.

I extend my appreciation to all my colleagues and fellow members of the Mathematics Department at Ferhat Abbas Setif 1 University, especially those in the LMFN Laboratory.

Finally, I would like to thank all those who contributed directly or indirectly to the elaboration of this work.

To all these people, THANK YOU.

OUAOUA MOHAMED LAMINE

Contents

List of Publications

Articles

1. Mohamed Lamine Ouaoua, Samia Khelladi and Djamel Benterki, *New parameter of conjugate gradient method for unconstrained nonlinear optimization*, **Statistics, Optimization & Information Computing**, 13(6), (2025), 0–8.
   https://doi.org/10.19139/soic-2310-5070-2069

2. Mohamed Lamine Ouaoua and Samia Khelladi, *Efficient Descent Direction of a Conjugate Gradient Algorithm for Nonlinear Optimization*, **Nonlinear Dynamics and Systems Theory**, 25(1) (2025) 91–100.
   https://www.e-ndst.kiev.ua/v25n1/9(97)a.pdf

List of Communications

International Communications

1. Ouaoua Mohamed Lamine, Khelladi Samia and Benterki Djamel, *Comparative study between different variants of Fletcher Reeves's conjugate gradient method for unconstrained optimization problems*, International Conference on Contemporary Mathematics and its Applications (ICCMA 2023), 26–27 November 2023, University Center Abdelhafid Boussouf Mila.
   https://icma23mila.sciencesconf.org/

2. Ouaoua Mohamed Lamine, Khelladi Samia and Benterki Djamel, *Comparative numerical study between different variants of conjugate gradient method of the Dai-Liao type*, International Conference on Mathematics and its Applications in Science and Technology (ICMAST'2024), 15–16 December 2024, Setif 1 University Ferhat Abbas.
   https://events.univ-setif.dz/event/6/

National Communications

1. Ouaoua Mohamed Lamine, *Étude numérique comparative des algorithmes de gradient conjugué basée sur deux nouvelles directions de descente*, Conférence Nationale Nouvelles Tendances en Mathématiques Théoriques et Computationnelles (CNNTMTC22), 8–9 Novembre 2022, Université de Tamanghasset.
   https://sites.google.com/view/hoggarmaths22/

2. Ouaoua Mohamed Lamine, Khelladi Samia and Benterki Djamel, *Étude comparative entre deux méthodes hybrides du gradient conjugué en utilisant différents types de recherche linéaire*, Conférence Nationale de Mathématiques et Applications (CNMA

2022), 29–30 Novembre 2022, Centre Universitaire Abdelhafid Boussouf Mila. https://sites.google.com/centre-univ-mila.dz/cnma2022-fr/

3. Ouaoua Mohamed Lamine, *Numerical study of two conjugate gradient methods for unconstrained optimization problem*, The National Conference On Mathematics & Applications (NCMA2023), 16 May 2023, Setif 1 University Ferhat Abbas. https://ocs.univ-setif.dz/NCMA23/NCMA23

Glossary of Abbreviations and Notations

Abbreviations

Abbreviation	Full Form	Definition
BFGS	Broyden-Fletcher-Goldfarb-Shanno	A quasi-Newton optimization algorithm
CD	Conjugate Descent	A conjugate gradient variant by Fletcher (1987)
CG	Conjugate Gradient	Optimization algorithm for linear/nonlinear systems
DFP	Davidon-Fletcher-Powell	An early quasi-Newton method
DY	Dai-Yuan	A conjugate gradient variant (1999)
FR	Fletcher-Reeves	The first nonlinear CG method (1964)
HS	Hestenes-Stiefel	The original CG method (1952)
HZ	Hager-Zhang	A CG variant with strong convergence (2006)
LS	Liu-Storey	A CG variant (1991)
PRP	Polak-Ribière-Polyak	A popular CG variant (1969)
QP	Quadratic Programming	Optimization with a quadratic objective function
UQP	Unconstrained Quadratic Problem	Minimization of quadratic functions

Mathematical Notations

Sets and Spaces

- $\mathbb{R}^n$: n-dimensional Euclidean space
- $\mathbb{N}$: Set of natural numbers
- Ω: Level set $\{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$
- $V(x_0)$: Neighborhood of point x_0

Functions and Operators

- $f(x)$: Objective function to minimize
- $\nabla f(x)$: Gradient of f at x
- $\nabla^2 f(x)$: Hessian matrix of f at x
- $\partial f(x)$: Subgradient of f at x
- $\langle x, y \rangle$: Inner product
- $\|x\|$: Euclidean norm
- $\mathcal{O}(\cdot)$: Big-O complexity

Algorithmic Components

- x_k: Iterate at step k
- d_k: Search direction at step k
- g_k: Gradient at x_k, i.e.,, $\nabla f(x_k)$
- α_k: step-size at iteration k
- β_k: CG parameter
- y_k: Gradient difference $g_{k+1} - g_k$
- s_k: Step difference $x_{k+1} - x_k$
- H_k: Hessian at point x_k

Convergence Conditions

- C1: Lipschitz continuity condition
- C2: Boundedness condition
- NC: Necessary Condition
- SC: Sufficient Condition
- NSC: Necessary and Sufficient Condition

Line Search Parameters

- ρ: Armijo condition parameter ($0 < \rho < 0.5$)
- σ: Wolfe condition parameter ($\rho < \sigma < 1$)
- δ, γ: Wu rule parameters
- μ: Parameter in hybrid methods ($\mu > 0$)
- ε: Convergence tolerance

Special Symbols

- $\arg\min$: Argument of the minimum
- $\liminf$: Limit inferior

CG Variant Parameters

Notation	Formula	Reference
β_k^{HS}	$\frac{g_k^\top y_{k-1}}{d_{k-1}^\top y_{k-1}}$	Hestenes-Stiefel (1952)
β_k^{FR}	$\frac{\|g_k\|^2}{\|g_{k-1}\|^2}$	Fletcher-Reeves (1964)
β_k^{PRP}	$\frac{g_k^\top y_{k-1}}{\|g_{k-1}\|^2}$	Polak-Ribière-Polyak (1969)
β_k^{DY}	$\frac{\|g_k\|^2}{d_{k-1}^\top y_{k-1}}$	Dai-Yuan (1999)
β_k^{DL}	$\frac{g_k^\top (y_{k-1} - t_k s_{k-1})}{d_{k-1}^\top y_{k-1}}$	Dai-Liao (2001)
β_k^{HZ}	$\frac{\left(y_{k-1} - \frac{2d_{k-1}\|y_{k-1}\|^2}{d_{k-1}^\top y_{k-1}}\right)^\top g_k}{d_{k-1}^\top y_{k-1}}$	Hager-Zhang (2006)
β_k^{RMIL}	$\frac{g_k^\top y_{k-1}}{\|d_{k-1}\|^2}$	Rivaie et al. (2012)
β_k^{MSD}	$\frac{\|g_k\|^2}{\|g_{k-1}\|^2 + \mu\|g_k^\top d_{k-1}\|}$	Ouaoua et al. (2025)
β_k^{OKB}	$\frac{\|g_k\|^2 - \|g_k\|\|d_{k-1}\|\|g_k^\top d_{k-1}\|}{d_{k-1}^\top y_{k-1}}$	Ouaoua et al. (2025)

List of Tables

List of Figures

Introduction

Optimization is a fundamental subject in applied mathematics and engineering that focuses on selecting the best possible solution from a set of available alternatives. It is widely and increasingly used across nearly every field of science and engineering, economics, management, industry. It deals with selecting the best of many possible decisions in real-life environment, constructing computational methods to find optimal solutions, exploring the theoretical properties, and studying the computational performance of numerical algorithms implemented based on computational methods, and solve problems of minimization or maximization [38]. The optimization of these systems makes it possible to find an ideal configuration and to achieve savings in effort, time, money, energy, raw materials, or even to increase satisfaction. Far from being an exhaustive list, these few examples demonstrate the variety of formulations and foreshadow the diversity of mathematical tools that may be used to solve such problems [23]. In general, optimization techniques play an increasingly important role in the design of systems and equipment of all kinds, as well as in all technical or economic decisions. Since people generally seek what is best, and when multiple options are available, their choice naturally turns to the most optimal one. Though optimization might date back to the very old extreme-value problems, it did not become an independent subject until the late, in 1940s George Dantzig, developed the simplex method that revolutionized linear programming by providing a systematic approach to navigating feasible solutions in multidimensional spaces. Initially applied to optimize U.S. Air Force logistics, it has since become a cornerstone in various industries, including finance and manufacturing. After the 1950s, when conjugate gradient methods and quasi-Newton methods were presented, the nonlinear programming developed greatly [38]. In 1952, Hestenes and Stiefel proposed conjugate gradient (CG) method [19] to solve linear systems.

Furthermore, Neculai Andrei [2] made significant contributions to optimization prob-

lems, its can be classified into both, constrained and unconstrained categories. This thesis focuses specifically on unconstrained optimization problems of the form:

$$(P)\begin{cases} \min f(x) \\ x \in \mathbb{R}^n \end{cases},$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function. While apparently simple, this formulation encompasses an enormous range of practical problems and presents significant theoretical and computational challenges, particularly when the dimension n becomes large.

In 1952, Hestenes and Stiefel [19] proposed conjugate gradient (CG) method to minimize convex quadratic functions which leads to the resolution of a linear system. Later, in 1964, Fletcher and Reeves [15] extended the method to solve nonlinear unconstrained optimization problems, then in 1969 by Polak, Ribière and Polyak [32, 33]. Since then, several variants of the conjugate gradient method have been proposed until our days.

As explained by Nocedal [29], the conjugate gradient method improves upon the basic gradient descent method by choosing search directions that reduce the zig-zag behavior common in older approaches. It is also memory-efficient, requiring only $\mathcal{O}(n)$ storage compared to the $\mathcal{O}(n^2)$ needed by Newton-based methods, making it suitable for solving large-scale problems.

The CG creates a sequence of points $\{x_k\}$ using the update rule:

$$x_{k+1} = x_k + \alpha_k d_k.$$

This means we move from the current point x_k in the direction d_k, by a certain amount α_k, called the *step-size*.

The direction d_k is chosen as follows:

$$d_k = \begin{cases} -g_k & \text{if } k = 1 \\ -g_k + \beta_k d_{k-1} & \text{if } k \geq 2. \end{cases}$$

Where $g_k = \nabla f(x_k)$ is the gradient of the function f at x_k. The gradient shows the direction of steepest descent, so $-g_k$ points in the direction of steepest decrease. $\alpha_k > 0$ is the step-size, calculated using a line search method. As for β_k, it is an important parameter that varies depending on the version of the method.

The choice of β_k has a big impact on both how fast the method converges and how well it works in practice. Over six decades of research have produced numerous β_k formulas, each with distinct characteristics:

- Fletcher-Reeves (FR) [15]: $\beta_k = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}$
- Polak-Ribière-Polyak (PRP) [33]: $\beta_k = \frac{g_k^T(g_k - g_{k-1})}{\|g_{k-1}\|^2}$
- Hager-Zhang (HZ) [18]: $\beta_k = \frac{(y_{k-1} - 2d_{k-1}\frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}})^T g_k}{d_{k-1}^T y_{k-1}}$
- Dai-Yuan (DY) [7]: $\beta_k = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}$,

where $y_{k-1} = g_k - g_{k-1}$.

Different versions of the method behave differently when it comes to convergence. The Fletcher-Reeves (FR) method is guaranteed to converge globally, but it may do so slowly. On the other hand, the Polak-Ribière-Polyak (PRP) method often works better in practice, even though its theoretical guarantees are not as strong [8].

Even with these improvements, nonlinear conjugate gradient methods still face important challenges. As pointed out by [18], some of the main issues are:

1. Finding a good balance between making enough progress (descent) and keeping directions properly related (conjugate)
2. Creating reliable ways to choose the step-size
3. Making sure the method works for all kinds of nonlinear functions (global convergence)
4. Keeping the method fast and efficient, especially in problems with many variables.

Later, many combinations and new families of conjugate gradient methods were proposed, in particular the hybrid methods. The idea of those methods is to combine the existent conjugate gradient methods in order to exploit the advantages of each of them to improve the performance of these methods and to avoid the jamming phenomenon. This combination can be convex or non-convex. Among these hybrid methods, we can cite those proposed by Dalladji et al. [10], Mtagulwa and Kaelo [27], Djordjevic [11, 12], Ben Hanachi et al. [3], the families of conjugate gradient methods proposed by Sellami and Chaib [36], Khelladi and Benterki [24, 25] and Hemici et al. [20].

The work of this thesis, makes a key contributions to CG methods, by given a new descent direction framework that strengthens the sufficient descent condition while preserving conjugacy. Also a two new proposed β_k parameters that adaptively balance conjugacy and descent properties.

This thesis contains an introduction, four chapters, and finally, a conclusion. In the first chapter, we start with the exposition of some preliminary and fundamental concepts of unconstrained nonlinear optimization, specifically, the existence and uniqueness results of an optimal solution, together with optimality conditions. Then, we introduce methods to solve unconstrained optimization problem, such as the gradient method, Newton method and the relaxation method. The second chapter presents the conjugate gradient method and its various variants with some global convergence results, before continuing on to present the main exact and inexact line search rules such as the Armijo, Wolfe, Backtracking, and Wu rules. The third chapter contains our first contribution, where we present a new descent direction of a conjugate gradient algorithm for nonlinear optimization followed by a theoretical analysis of global convergence and numerical tests. The fourth chapter contains our second contribution. In fact, we present a new approach based on a second new parameter of a conjugate gradient algorithm for unconstrained nonlinear optimization with a theoretical analysis of global convergence and numerical tests. Lastly, we conclude this work with a conclusion and then a bibliography.

Chapter 1

Preliminaries on the unconstrained optimization

In this chapter, we survey concepts related to unconstrained optimization that we will refer to during our work, along with some useful properties.

1.1 Unconstrained nonlinear optimization

1.1.1 Concepts and definitions

The optimization problem serves to find a minimum or maximum of some given quantity, which is referred to as the objective.

Definition 1.1.1. *An unconstrained optimization problem is simply a problem which has the form of a minimization or a maximization problem:*

$$(P)\begin{cases}\min f(x)\\ x\in\mathbb{R}^n\end{cases} \quad or\ (P')\begin{cases}\max f(x)\\ x\in\mathbb{R}^n\end{cases},$$

where $f:\mathbb{R}^n\to\mathbb{R}$ is a continuous function ($f\in\mathcal{C}(\mathbb{R}^n)$).

f is called the "objective" or "economic" function.

An optimal solution of (P) (or (P')) is any point $x^*\in\mathbb{R}^n$ satisfying: $f(x^*)=\min\limits_{\mathbb{R}^n} f(x)$, (resp. $f(x^*)=\max\limits_{\mathbb{R}^n} f(x)$).

In this case, $f(x^*)=f^*$ is called the optimal value of f.

Remark 1.1.2. *1)* $\max f(x)=-\min(-f(x))$, $\forall x\in\mathbb{R}^n$.

So, we have equivalencies between (P) and (P'), which means that whatever method we use to solve (P) can also be used to solve (P'). Therefore, we are only concerned with studying (P).

2) Solving the following problem

$$(P)\begin{cases}\min f(x)\\ x\in\mathbb{R}^n\end{cases},$$

amounts to:

i) Finding an optimal solution $x^*\in\mathbb{R}^n$ of (P).

ii) If we cannot find a solution x^*, we proceed to find an approximate optimal value of f^*.

Definition 1.1.3. *A set $D\subseteq\mathbb{R}^n$ is called **convex** if for any two points $x,y\in D$ and any $\lambda\in[0,1]$, the point $\lambda x+(1-\lambda)y$ also lies within D. Formally,*

$$\lambda x+(1-\lambda)y\in D,\quad \forall x,y\in D,\ \forall\lambda\in[0,1].$$

Definition 1.1.4. *Let $f:D\subset\mathbb{R}^n\to\mathbb{R}$ and D be a convex set in $\mathbb{R}^n$.*

*1) We say that f is **convex** if and only if for all $x,y\in D$, we have:*

$$f[(1-t)x+ty]\leq(1-t)f(x)+tf(y),\ \text{for all } t\in[0,1]\,.$$

*2) We say that f is **strictly convex** if and only if for all $x,y\in D$, with $x\neq y$, we have:*

$$f[(1-t)x+ty]<(1-t)f(x)+tf(y),\ \text{for all } t\in\,]0,1[\,.$$

*3) We say that f is **strongly convex** with modulus $\delta>0$ (δ-convex), if for all $x,y\in D$, we have:*

$$f[(1-t)x+ty]\leq(1-t)f(x)+tf(y)-\frac{\delta}{2}t(1-t)\left\|x-y\right\|^2,\ \text{for all } t\in]0,1[,$$

where $\|\cdot\|$ is the Euclidean norm.

Definition 1.1.5. *Let be the problem:*

$$(P)\begin{cases} \min f(x) \\ x \in \mathbb{R}^n \end{cases},$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $f \in \mathcal{C}(\mathbb{R}^n)$. *We have:*

1) $x^* \in \mathbb{R}^n$ *is* ***a local minimum*** *of* (P) *if and only if:*

$$\exists \alpha > 0 \text{ such that } f(x^*) \leq f(x) : \forall x \in B(x^*, \alpha), \tag{1.1}$$

where

$$B(x^*, \alpha) = \{x \in \mathbb{R}^n / \, \|x - x^*\| < \alpha\}.$$

The condition (1.1) holds only in a neighborhood of x^*.

2) $x^* \in \mathbb{R}^n$ *is* ***a global minimum*** *of* (P) *if and only if:*

$$f(x^*) \leq f(x) : \forall x \in \mathbb{R}^n.$$

3) The set of all minimizers of (P) *is noted by* $\arg\min_{\mathbb{R}^n} f(x)$, *i.e.,*

$$x^* \in \arg\min_{\mathbb{R}^n} f(x) \Leftrightarrow f(x^*) = \min_{\mathbb{R}^n} f(x).$$

Remark 1.1.6. *1)* ($x^* \in \mathbb{R}^n$ *is a global minimum of* f) $\Rightarrow$ ($x^* \in \mathbb{R}^n$ *is a local minimum of* f).

The converse is not true in general.

2) If f *is convex then every local minimum is a global minimum, i.e.,*

(x^* *is a global minimum of* f) $\Leftrightarrow$ (x^* *is a local minimum of* f).

Definition 1.1.7. *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *such that* $f \in \mathcal{C}(\mathbb{R}^n)$.

1) If f *is differentiable at* $x_0 \in \mathbb{R}^n$, *then the derivative of* f *at* x_0 *is a vector which is called* ***gradient****, and is denoted* $\nabla f(x_0)$ *as:*

$$\nabla f(x_0) = \left(\frac{\partial f}{\partial x_1}(x_0), \ldots, \frac{\partial f}{\partial x_n}(x_0)\right)^T,$$

where $\dfrac{\partial f}{\partial x_i}(x_0)$ *denotes the partial derivative of* f *with respect to the* i*-th variable of* x.

2) If f *is twice differentiable at* $x_0 \in \mathbb{R}^n$, *then the second derivative of* f *at* x_0 *is the*

***Hessian matrix**, which is denoted as $H(x_0)$ (or $\nabla^2 f(x_0)$) which is an $n \times n$ matrix, defined by:*

$$(H(x_0))_{ij} = (\nabla^2 f(x_0))_{ij} = \left(\frac{\partial^2 f}{\partial x_i \partial x_j}\right)(x_0),\ i = 1, \ldots, n,\ j = 1, \ldots, n,$$

where $\dfrac{\partial^2 f}{\partial x_i \partial x_j}$ denotes the second partial derivative of f with respect to the i-th and j-th variable of x.

Remark 1.1.8. *If f is convex and non-differentiable at a point $x_0 \in \mathbb{R}^n$, then we will use the subgradient ($g_0 \in \partial f(x_0)$) of f at that point x_0 instead of its gradient $\nabla f(x_0)$, where*

$$\partial f(x_0) = \left\{ g \in \mathbb{R}^n \;\middle|\; f(x) \geq f(x_0) + g^T(x - x_0),\ \forall x \in \mathbb{R}^n \right\}.$$

1.1.2 Existence and uniqueness results

Before exploring the characteristics of any solutions to the problem (P) it's necessary to confirm whether such solutions exist.

Definition 1.1.9. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function. f is **coercive** if and only if:*

$$\lim_{\|x\| \to +\infty} f(x) = +\infty.$$

In other words:

$$\forall A \in \mathbb{R}, \exists B > 0 \text{ such that } \forall x \in \mathbb{R}^n, \| x \| \geq B \Longrightarrow f(x) \geq A.$$

In a given situation, we select the norm that is most appropriate for the function under consideration.

Remember that all norms in $\mathbb{R}^n$ are equivalent. We have the following standard forms:

$$\forall x \in \mathbb{R}^n, p \in \mathbb{N}^* : \| x \|_p = \left[\sum_{i=1}^{n} |x_i|^p \right]^{\frac{1}{p}}.$$

If $p = 2$, then $\| \cdot \|_2$ is the Euclidean norm of $\mathbb{R}^n$.

The infinity norm on $\mathbb{R}^n$ is defined as follows:

$$\forall x \in \mathbb{R}^n, \| x \|_\infty = \max_{1 \leq i \leq n} \{|x_i|\}.$$

Theorem 1.1.10 (Weierstrass). *[23] Let $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be a continuous function on a compact set D (i.e., closed and bounded). Then, f has at least one minimum on D.*

i.e., $\exists x^* \in D$ such that $f(x^*) = \min_{x \in D} f(x) \iff \exists x^* \in D$ such that $f(x^*) \leq f(x), \forall x \in D$.

Theorem 1.1.11 (Existence). *[23] If $f : \mathbb{R}^n \to \mathbb{R}$ is continuous and coercive on $\mathbb{R}^n$, then f has at least one minimum x^* on $\mathbb{R}^n$,*

i.e., $\exists x^* \in \mathbb{R}^n$ such that $f(x^*) \leq f(x), \forall x \in \mathbb{R}^n$.

Theorem 1.1.12 (Sufficient condition for uniqueness). *[23] If $f : \mathbb{R}^n \to \mathbb{R}$ is strictly convex, then there exists at most one $x^* \in \mathbb{R}^n$ such that:*

$$f(x^*) \leq f(x), \forall x \in \mathbb{R}^n.$$

Theorem 1.1.13 (Existence and uniqueness). *Let $f : \mathbb{R}^n \to \mathbb{R}$. Suppose that f is continuous, coercive and strictly convex.*

Then there exists a unique $x^* \in \mathbb{R}^n$ such that $f(x^*) = \min f(x)$, and the problem (P) admits a unique global optimal solution x^*.

1.1.3 Optimality conditions

Necessary conditions (NC)

Theorem 1.1.14 (First-order NC). *[23] Let $f : \mathbb{R}^n \to \mathbb{R}$, continuous and differentiable at $x^* \in \mathbb{R}^n$. If x^* is a minimum (global or local) of the function f, then $\nabla f(x^*) = 0$.*

Theorem 1.1.15 (Second-order NC). *[23] Let $f : \mathbb{R}^n \to \mathbb{R}$, continuous and twice differentiable at $x^* \in \mathbb{R}^n$. If x^* is a minimum (global or local) of the function f, then:*

1) $\nabla f(x^*) = 0$.

2) The Hessian matrix $H(x^*)$ of f at the point x^* is positive semidefinite:

$$\langle H(x^*)d, d\rangle = d^T H(x^*)d \geq 0, \forall d \in \mathbb{R}^n.$$

i.e.,, f is convex in the neighborhood of x^*.

Remark 1.1.16. *1) For the unidimensional case (i.e., $f : \mathbb{R} \to \mathbb{R}$), we have:*

$$(x^* \in \mathbb{R} \text{ minimum of } f) \Rightarrow \begin{cases} f'(x^*) = 0 \\ f''(x^*) \geq 0 \end{cases}.$$

*2) A point $x^* \in \mathbb{R}^n$ satisfying $\nabla f(x^*) = 0$, is called **stationary** (or **critical** point) of f in $\mathbb{R}^n$ (a minimum candidate).*

But, $\nabla f(x^*) = 0 \not\Rightarrow x^*$ is a minimum of f.

3) If f is differentiable and $\nabla f(x^*) \neq 0$, then x^* cannot be a minimum of f.

Sufficient conditions (SC)

Theorem 1.1.17. *[23] Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice differentiable at $x^* \in \mathbb{R}^n$, if:*

1) $\nabla f(x^*) = 0$.

2) $H(x^*)$ is positive definite:

$$\langle H(x^*)d, d\rangle = d^T H(x^*)d > 0, \forall d \in \mathbb{R}^n.$$

Then x^* is a local minimum of f.

Necessary and sufficient conditions (NSC)

Theorem 1.1.18. *[23] Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable and convex function on $\mathbb{R}^n$, then:*

$$x^* \in \mathbb{R}^n \text{ is a global minimum of } f \text{ if and only if } \nabla f(x^*) = 0.$$

Remark 1.1.19. *1) If f is convex and non-differentiable at x^* then NSC becomes:*

x^* is a minimum of $f \Leftrightarrow 0 \in \partial f(x^*)$.

2) Therefore, differentiability and convexity matter in solving optimization problems. In the absence of differentiability or convexity, the study of the problem becomes more complicated.

1.2 Descent methods

1.2.1 Main idea of descent methods

The idea of descent methods is to find a vector x^* which minimize the function f. For this, we take some arbitrary initial point x_1. To build the second iterate x_2, we should move in the direction of the minimum x^* of f. So, we need to guarantee that $f(x_2) \leq f(x_1)$.

We are now looking for x_2 in the form: $x_2 = x_1 + \alpha_1 d_1$ where $\alpha_1 \in \mathbb{R}_+^*$ is a step-size and $d_1 \in \mathbb{R}^n$ $(d_1 \neq 0)$ is a descent direction chosen in such a way that $f(x_1 + \alpha_1 d_1) \leq f(x_1)$. This process goes on and generates a sequence (x_k) beginning at x_1 and converging to x^* (with x^* a stationary point of f, i.e., $\nabla f(x^*) = 0$). Hence, the general scheme of a descent method is given as:

$$\begin{cases} x_1 \in \mathbb{R}^n \text{ (Initial point)} \\ x_{k+1} = x_k + \alpha_k d_k \end{cases}.$$

Recall that by definition d_k is a descent direction of f at point x_k if:

$$f(x_{k+1} + \alpha_k d_k) \leq f(x_k).$$

Proposition 1.2.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable on $\mathbb{R}^n$, then d_k is a descent direction of f at point x_k if and only if:*

$$\langle \nabla f(x_k), d_k \rangle = \nabla f(x_k)^T d_k < 0.$$

Remark 1.2.2. *1) The step-size can either be fixed or adjusted at each iteration.*

2) If the direction d_k is chosen based on $\nabla f(x_k)$, the method is said to be of order 1.

3) If the direction d_k is chosen based on the second derivatives (i.e., the Hessian matrix $H(x_k)$), the method is said to be of order 2.

Among the descent methods, that we will cover we have **the gradient method**, **Newton's method**, **quasi-Newton methods**, **the relaxation method**, which will be presented bellow, and **the conjugate gradient method** which will be discussed in detail in the second chapter.

1.2.2 Gradient method

The idea of the method goes back to **Cauchy** in 1847. It is a natural concept based on the first-order Taylor expansion of the function f between two iterations x_k and $x_{k+1} = x_k + \alpha_k d_k$.

We have:

$$f(x_{k+1}) = f(x_k + \alpha_k d_k) = f(x_k) + \alpha_k \langle \nabla f(x_k), d_k \rangle + O(\alpha_k d_k).$$

Since we want that $f(x_{k+1}) < f(x_k)$, we can choose $d_k = -\nabla f(x_k)$.

This gives us:

$$\begin{aligned} f(x_k + \alpha_k d_k) - f(x_k) &= \alpha_k \langle \nabla f(x_k), -\nabla f(x_k) \rangle + O(\alpha_k d_k) \\ &= -\alpha_k \left\| \nabla f(x_k) \right\|^2 + O(\alpha_k d_k) < 0. \end{aligned}$$

The method thus obtained is called the gradient method.

Moreover, the choice of the step-size α_k can be done as follows:

1. Let $\alpha_k = \alpha > 0$ be fixed. This is called the fixed step-size or constant step-size gradient method.

2. Let α_k be chosen as the minimum of the function $\varphi(\alpha) = f(x_k - \alpha \nabla f(x_k))$. This is called the gradient method with optimal step-size.

3. The step α_k can be varied at each iteration (example: $\alpha_k = \frac{1}{k}$). This is called the divergent series variable step-size gradient method.

Algorithm Gradient Method

Begin algorithm

- Given a starting point $x_1 \in \mathbb{R}^n$, $\varepsilon > 0$ and $k = 1$.
- While $\|\nabla f(x_k)\| \geq \varepsilon$ do
  - Compute $d_k = -\nabla f(x_k)$.
  - Choose a step-size $\alpha_k > 0$ such that $f(x_k + \alpha_k d_k) < f(x_k)$.

  - Update $x_{k+1} = x_k + \alpha_k d_k$.
  - Set $k = k + 1$.

- End while.

End algorithm.

Remark 1.2.3. *1) Other stopping criteria that we can select from include:*

- $\|x_{k+1} - x_k\| < \varepsilon$.
- $|f(x_{k+1}) - f(x_k)| < \varepsilon$.

2) Furthermore, since convergence is not always guaranteed, we can set a maximum number of iterations $k_{\max}$.

One of the advantages of the gradient method is its ease of implementation. Unfortunately, the method is generally slow (linear convergence) and the convergence conditions are very strict (essentially, need strict convexity). This is because of the zigzag phenomenon, which occurs when successive directions d_k and d_{k+1} are orthogonal. We give a convergence criterion below.

Theorem 1.2.4. *[23] Let* $f : \mathbb{R}^n \to \mathbb{R}$ *be a* $\mathcal{C}^1$ *function and* x^* *be a minimum of* f. *Assume that:*

1. f *is* β*-elliptic, i.e.,:*

$$\exists \beta > 0, \forall x, y \in \mathbb{R}^n : \langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \beta \|x - y\|^2 .$$

2. *The gradient* ∇f *is Lipschitz continuous, i.e.,:*

$$\exists M > 0, \forall x, y \in \mathbb{R}^n : \|\nabla f(x) - \nabla f(y)\| \leq M \|x - y\|.$$

Then, the gradient method then converges (for any choice of x_1*) to the minimum of* f *in a linear order if we select the step-size* α_k *such that* $0 < \alpha_k < \frac{2\beta}{M}$, *for all* $k \geq 0$ *i.e.,*

$$\exists C \in]0, 1[\text{ such that } \|x_{k+1} - x^*\| \leq C \|x_k - x^*\| \leq C^k \|x_1 - x^*\| .$$

Remark 1.2.5. *1) If f is β-elliptic, then f is strictly convex and coercive.*

2) In practice, we do not calculate β and M. To find the interval of convergence for the step-sizes α_k, several tests are performed for different values. Non-convergence usually occurs either in explosion behavior of the solution (converging to $+\infty$ certainly) or in oscillations (cyclic or not) which do not allow the succession of iterates to converge to a point.

1.2.3 Newton method

We assume that the function f is twice continuously differentiable ($f \in \mathcal{C}^2(\mathbb{R}^n)$) and that we can compute all of its second-order derivatives.

The Newton method aims to approximate the function f in the neighborhood of the current point x_k by its quadratic approximation q (using the Taylor expansion up to second order in the neighborhood of x_k), we obtain:

$$f(x) = q(x) \simeq f(x_k) + \langle x - x_k, \nabla f(x_k)\rangle + \frac{1}{2}\langle H(x_k)(x - x_k), x - x_k\rangle,$$

where $\nabla f(x_k)$ is the gradient of f at x_k and $H(x_k)$ is the Hessian matrix of f at x_k.

We then take x_{k+1} as the minimum of $q(x)$ if it exists. This will only hold when $H(x_k)$ is a positive definite matrix. Then $q(x)$ is strictly convex and admits a unique minimum x_{k+1} such that: $\nabla q(x_{k+1}) = 0$.

This gives us the following linear system:

$$\begin{aligned} H(x_k)(x_{k+1} - x_k) &= -\nabla f(x_k). \\ &\Leftrightarrow H(x_k)d_k = -\nabla f(x_k) \text{ where } d_k = x_{k+1} - x_k. \\ &\Leftrightarrow d_k = -H^{-1}(x_k)\nabla f(x_k). \end{aligned}$$

Let $H_k = H(x_k)$ such that H_k is a positive definite matrix and we define d_k as the Newton direction, which derives the iteration formula:

$$x_{k+1} = x_k - H_k^{-1}\nabla f(x_k). \tag{1.2}$$

So, for a $\mathcal{C}^2-$function $f : \mathbb{R}^n \to \mathbb{R}$, we get the following algorithm:

Newton Algorithm

Begin algorithm

- Given a starting point x_1 in a neighborhood of x^*, $\varepsilon > 0$ and $k = 1$.
- While $\|\nabla f(x_k)\| \geq \varepsilon$ do
  - Solve the linear system $H_k d_k = -\nabla f(x_k)$ to obtain d_k.
  - Update $x_{k+1} = x_k + d_k$.
  - Set $k = k + 1$.
- End while.

End algorithm.

Remark 1.2.6. *1) The formula (1.2) is the Newton's iterative scheme used to solve the system* $\nabla f(x_k) = 0$.

2) The Newton's method is a descent algorithm with a fixed step-size $\alpha = 1$.

3) The Newton's algorithm converges in a single iteration regardless of the starting point x_1 *if the function* f *is quadratic and strictly convex.*

However, the algorithm may diverge for an arbitrary function if the initial point x_1 *is not close enough to the minimum* x^* *or if* H^{-1} *does not exist.*

For the case of a strictly convex quadratic function, we have the following theorem on the convergence of Newton method.

Theorem 1.2.7. *[23] Let* Q *be a symmetric, positive definite* $n \times n$ *matrix,* $x \in \mathbb{R}^n, b \in \mathbb{R}^n$. *Consider the unconstrained quadratic problem* (UQP)*:*

$$(UQP) : \min \left\{ \frac{1}{2} x^T Q x - b^T x;\ x \in \mathbb{R}^n \right\},$$

and denote x^* *to be the globally optimal solution of* (UQP) $(x^* = Q^{-1}b)$.

Let $\{x_k\}$ *be the sequence generated by Newton method, then* $\{x_k\}$ *converges to* x^* *in one iteration.*

Remark 1.2.8. *Newton method has the great advantage in speed of convergence. But it has two key disadvantages:*

1. *Local convergence: It is dependent on the initial point x_1 choice; the method may diverge if $x_1 \notin V(x^*)$ (x_1 is not in the neighberhood of x^*).*

2. *Costly computation of the direction d_k: This involves computing $\nabla f(x_k)$, H_k, and solving the system $H_k d_k = -\nabla f(x_k)$ or finding H_k^{-1}.*

To mitigate these downsides, some variants are developed, known as quasi-Newton methods*, such as* ***the Levenberg-Marquardt method***, ***Davidon-Fletcher-Powell (DFP) method***, *and* ***Broyden-Fletcher-Goldfarb-Shanno (BFGS) method***.

1.2.4 Quasi-Newton methods

A quasi-Newton method is formulated as:

$$\begin{cases} d_k = -M_k \nabla f(x_k) \\ x_{k+1} = x_k + \alpha_k d_k \end{cases} \quad \text{for } k \geq 1,$$

with step-size α_k being given by a line search in direction d_k and M_k being a matrix built to approximate the inverse of the Hessian of f, which is symmetric and positive definite. This raises the problem: how to approximate this inverse?

For example, we can start with $M_0 = I$, but how do we update the approximation M_k during the iteration?

The reasoning goes like this: we know at point x_k the gradient and the Hessian of f are related by the following equation:

$$\nabla f(x) = \nabla f(x_k) + H(x_k)(x - x_k) + o(\|x - x_k\|).$$

In other words:

$$\nabla f(x) = \nabla f(x_k) + H(x_k)(x - x_k), \ x \in V(x_k).$$

Or alternatively:

$$H^{-1}(x_k)(\nabla f(x) - \nabla f(x_k)) = (x - x_k).$$

Indeed, if f is quadratic, these approximations are exact. In particular, for $x = x_{k+1}$

and if M_k is a good approximation of $H^{-1}(x_k)$, we should have:

$$M_k(\nabla f(x_{k+1}) - \nabla f(x_k)) = (x_{k+1} - x_k).$$

However, because x_{k+1} is computed after M_k, this equation is unlikely to hold, at least not approximately. But we can always impose that M_{k+1} holds exactly to this equation, so:

$$M_{k+1}(\nabla f(x_{k+1}) - \nabla f(x_k)) = (x_{k+1} - x_k).$$

Let define:

$$\begin{aligned} s_k &= x_{k+1} - x_k, \\ y_k &= \nabla f(x_{k+1}) - \nabla f(x_k) \end{aligned},$$

hence:

$$M_{k+1} y_k = s_k.$$

This equation is called "**secant equation**" or *"**quasi-Newton condition**"*.

Davidon-Fletcher-Powell method (DFP)

This is the earliest known application of quasi-Newton methods that sought to optimize the function f. The DFP method approximates the Hessian matrix by using the matrix M_k in place of its inverse. The average symmetry of these approximations is maintained through a very simple form of updating:

$$M_{k+1} = M_k + C_k.$$

Where C_k is a rank-2 matrix such that:

$$C_k = a_k u_k u_k^T + b_k v_k v_k^T.$$

where a_k, b_k are constants, and u_k, v_k are two vectors in $\mathbb{R}^n$.

The matrix M_{k+1} must satisfy the quasi-Newton condition, i.e.,:

$$s_k = M_{k+1} y_k.$$

Thus, we obtain the DFP formula:

$$M_{k+1}^{DFP} = M_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{M_k y_k y_k^T M_k}{y_k^T M_k y_k}.$$

Remark 1.2.9. *The DFP method works sufficiently and in practice is superseded by* ***BFGS*** *method which is more effective.*

Broyden-Fletcher-Goldfarb-Shanno method (BFGS)

This method has been developed as the independent work of **Broyden**, **Fletcher**, **Goldfarb**, and **Shanno**.

The formula **BFGS** is a correction rank-2 that directly becomes from the formula **DFP**:

$$M_{k+1} = M_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{M_k y_k y_k^T M_k}{y_k^T M_k y_k}.$$

After manipulating over this expression, Fletcher observes that upon interchanging the roles of s_k and y_k in the DFP formula and switching to the following sequence of matrix:

$$\begin{cases} B_1 \text{ symmetric positive definite matrix} \\ B_{k+1} = B_k + \dfrac{y_k y_k^T}{y_k^T s_k} - \dfrac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} \end{cases},$$

then the obtained matrices satisfy the 'inverse' relation of the quasi-Newton condition, that is:

$$B_{k+1} s_k = y_k,$$

and this allows for the formulation of an approximation B_k of the Hessian H instead of its inverse as in the DFP method.

To derive a correction formula to estimate the inverse of the Hessian (H^{-1}), it is sufficient to take inverse on both sides of the formula for B_{k+1}.

After manipulating over this expression, and by setting $M_k = B_k^{-1}$ and $M_{k+1} = B_{k+1}^{-1}$, we end up with BFGS which gives the approximate value of the inverse Hessian of function f as follows:

$$M_{k+1}^{BFGS} = M_k + \left[1 + \frac{y_k^T M_k y_k}{s_k^T y_k}\right] \frac{s_k s_k^T}{s_k^T y_k} - \frac{s_k y_k^T M_k + M_k y_k s_k^T}{s_k^T y_k}.$$

1.2.5 Relaxation method

Let be the following problem:

$$(P)\begin{cases} \min f(x) \\ x \in \mathbb{R}^n, \end{cases}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $x = (x^1, x^2, \ldots, x^n) \in \mathbb{R}^n$.

The relaxation method allows reducing a minimization problem in $\mathbb{R}^n$ to the successive resolution of n minimization problems in $\mathbb{R}$ at each iteration.

The method relies on the following principle:

For an iterate x_k, we fix all but the first components and minimize the function f respect to the first component, i.e.,:

$$\min f(t, x_k^2, \ldots, x_k^n) \text{ for } t \in \mathbb{R}.$$

This yields the first coordinate of the next iterate, x_{k+1} denoted x_{k+1}^1. Minimization of this function in $\mathbb{R}$ can be done using methods such as Newton's method.

Next, we keep the first coordinate x_{k+1}^1 and the $n-2$ coordinates the same as before, and minimize with respect to the second coordinate, i.e.,:

$$x_{k+1}^2 = \arg\min f(x_{k+1}^1, t, x_k^3, \ldots, x_k^n),$$

and so on.

The resulting algorithm is:

Relaxation Algorithm

Begin algorithm

- Given a starting point $x_1 \in \mathbb{R}^n$, $\varepsilon > 0$ (precision) and $k = 1$.
- While $\|\nabla f(x_k)\| \geq \varepsilon$ do
  - For $i = 1$ to n, compute x_{k+1}^i as the solution of

$$\min_{t \in \mathbb{R}} f(x_{k+1}^1, \ldots, x_{k+1}^{i-1}, t, x_k^{i+1}, \ldots, x_k^n).$$

  - Set $k = k + 1$.

- End while.

End algorithm.

In the next chapter, we present a detailed study of conjugate gradient method and its various approaches well known in the literature.

Chapter 2

Conjugate gradient method and line search techniques

The conjugate gradient method is a numerical algorithm used to solve systems of linear equations $Qx = b$, in which the matrix Q is positive definite. **Cornelius Lanczos** (a Hungarian), **Magnus Hestenes** (an American), and **Edward Stiefel** (a Swiss) all developed this technique simultaneously in 1950 and 1951. This approach is iterative and converges in a finite number of iterations, which is equal to the system's size.

This method is also used for solving unconstrained optimization problems. It is a descent method that uses conjugate directions, which are not orthogonal, to overcome the gradient method's drawback of orthogonal directions. It can therefore be categorized between Newton method and the gradient method.

2.1 Conjugate gradient method

Conjugate gradient methods are based on the conjugate directions principle.

Definition 2.1.1 (Q-conjugate vectors)**.** *Consider a symmetric positive definite matrix $Q \in M_n(\mathbb{R})$. The set of vectors $\{d_1, d_2, \ldots, d_k\}$ of $\mathbb{R}^n \setminus \{0\}$ is considered Q-conjugate if:*

$$(d_j)^T Q d_i = \langle Q d_i, d_j \rangle = 0, \quad \forall i \neq j \text{ and } 1 \leq i \leq k, 1 \leq j \leq k.$$

It means that two Q-conjugate vectors are orthogonal with respect to the inner product associated with matrix Q, which is defined as follows: $\langle x, y \rangle_Q = y^T Q x, \forall x, y \in \mathbb{R}^n$.

2.1.1 Conjugate gradient method: Linear case

Consider a symmetric positive definite matrix $Q \in M_n(\mathbb{R})$. Examine the quadratic problem (QP):

$$(QP) \begin{cases} \min f(x) = \frac{1}{2}x^T Q x - b^T x \\ x \in \mathbb{R}^n. \end{cases}$$

For the linear case, the conjugate gradient method is an iterative process. The sequence $\{x_k\}$ is generated as follows:

1) An initial vector x_1 serves as our starting point.

2) The successor x_{k+1} of x_k at each step k is determined by:

$$\begin{cases} x_1 \in \mathbb{R}^n \text{ (initial point)} \\ x_{k+1} = x_k + \alpha_k d_k \quad k \geq 1 \end{cases}, \tag{2.1}$$

where d_k is the descent direction and α_k is the step-size.

The recurrence formula defines the descent directions d_k is:

$$d_k = \begin{cases} -g_1 & \text{if } k = 1 \\ -g_k + \beta_k d_{k-1} & \text{if } k \geq 2 \end{cases} \quad \text{where } g_k = \nabla f(x_k). \tag{2.2}$$

The step-size α_k is determined using an exact or inexact line search, and the scalar $\beta_k \in \mathbb{R}$ is selected so that d_k is Q-conjugate with the earlier directions d_i, $i = 1, \ldots, k-1$. In other words, we require:

$$d_k^T Q d_i = 0, \quad \forall i = 1, \ldots, k-1.$$

The name "conjugate gradient" combines two terms: "gradient" and "conjugate".

a) The word "gradient" is used because d_k is computed from the gradient at the point x_k.

b) The word "conjugate" describes the conjugate directions $\{d_k \mid k \geq 1\}$, not the gradients $\{g_k \mid k \geq 1\}$. Thus, "conjugate directions method" would be a more appropriate name.

Determination of the step-size α_k

The step-size α_k can be found by solving the following one-dimensional optimization problem:

$$\alpha_k = \arg\min f(x_k + \alpha d_k), \quad \alpha > 0.$$

Let $\varphi_k(\alpha) = f(x_k + \alpha d_k)$. Since α_k minimizes f along the direction d_k, it satisfies the optimality condition:

$$\varphi_k'(\alpha_k) = d_k^T \nabla f(x_{k+1}) = 0.$$

This leads to:

$$d_k^T \nabla f(x_{k+1}) = d_k^T (Qx_{k+1} - b) = 0,$$

which gives:

$$d_k^T Q(x_k + \alpha_k d_k) - d_k^T b = 0.$$

Simplifying, we obtain:

$$\alpha_k = -\frac{g_k^T d_k}{d_k^T Q d_k}. \tag{2.3}$$

Determination of the coefficient β_k

The coefficients β_k are chosen so that d_k is conjugate to all previous directions. Specifically:

$$d_k^T Q d_{k-1} = 0.$$

Using (2.2), we get:

$$(-g_k + \beta_k d_{k-1})^T Q d_{k-1} = 0.$$

Solving for β_k, we obtain:

$$\beta_k = \frac{g_k^T Q d_{k-1}}{d_{k-1}^T Q d_{k-1}}.$$

Conjugate gradient algorithm (linear case)

Begin algorithm

- Given an initial point $x_1 \in \mathbb{R}^n$, $\varepsilon > 0$ and $k = 1$.
- Compute $g_1 = Qx_1 - b$ and set $d_1 = -g_1$.
- While $\|g_k\| \geq \varepsilon$ do

- Compute $\alpha_k = -\dfrac{g_k^T d_k}{d_k^T Q d_k}$.
- Update $x_{k+1} = x_k + \alpha_k d_k$.
- Compute $g_{k+1} = Qx_{k+1} - b$.
- Compute $\beta_{k+1} = \dfrac{g_{k+1}^T Q d_k}{d_k^T Q d_k}$.
- Set $d_{k+1} = -g_{k+1} + \beta_{k+1} d_k$.
- Set $k = k + 1$.

- End while.

End algorithm.

Theorem 2.1.2. *[23] For any $x_1 \in \mathbb{R}^n$, the sequence $\{x_k\}$ generated by the iterative scheme* (2.1) *and* (2.2) *converges to the solution x^* of the linear system $Qx = b$ in at most n iterations.*

2.1.2 Conjugate gradient method: Nonlinear case

General principle

The linear conjugate gradient method is mainly used for solving linear systems with a symmetric positive definite matrix. A natural question is whether this method can be extended to solve nonlinear optimization problems.

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a non-quadratic function. Consider the nonlinear optimization problem:

$$(P) \begin{cases} \min f(x) \\ x \in \mathbb{R}^n \end{cases}.$$

The goal is to iteratively construct mutually conjugate directions $d_1, \ldots, d_k$. At each iteration k, the direction d_k is obtained by a linear combination of the gradient of f at x_k and the previous direction d_{k-1}. The combination is chosen to ensure d_k is conjugate to all previous directions, making the method effective for minimizing the function.

The direction d_k is defined by the recurrence relation:

$$d_k = \begin{cases} -g_k & \text{if } k = 1 \\ -g_k + \beta_k d_{k-1} & \text{if } k \geq 2 \end{cases},$$

where $g_k = \nabla f(x_k)$, $\beta_k \in \mathbb{R}$, and $\{x_k\}_{k\in\mathbb{N}}$ is generated by (2.1).

The step-size $\alpha_k \in \mathbb{R}$ is determined by an exact or inexact line search.

Different variants of this method extend the conjugate gradient method to the non quadratic case. Notable contributions include the **Fletcher-Reeves (FR)** method (1964)[15], the **Polak-Ribière-Polyak (PRP)** method (1969)[32, 33], and the **Dai-Yuan (DY)** method (1999)[7]. Many other variants have been proposed until our days. In the next table we cite some of the classical parameters of β_k. For other parameters of β_k, see [2].

Table 2.1: List of some variants of the nonlinear conjugate gradient method

β_k Formula	C.G. Variant	Date
$\beta_k^{HS} = \dfrac{y_{k-1}^T g_k}{y_{k-1}^T d_{k-1}}$	Hestenes-Stiefel	1952
$\beta_k^{FR} = \dfrac{\|g_k\|^2}{\|g_{k-1}\|^2}$	Fletcher-Reeves	1964
$\beta_k^{D} = \dfrac{g_k^T \nabla^2 f(x_k) d_k}{d_k^T \nabla^2 f(x_k) d_k}$	Daniel	1967
$\beta_k^{PRP} = \dfrac{y_{k-1}^T g_k}{\|g_{k-1}\|^2}$	Polak-Ribière-Polyak	1969
$\beta_k^{CD} = -\dfrac{\|g_k\|^2}{g_{k-1}^T d_{k-1}}$	Conjugate descent	1987
$\beta_k^{LS} = -\dfrac{y_{k-1}^T g_k}{g_{k-1}^T d_{k-1}}$	Liu-Storey	1991
$\beta_k^{DY} = \dfrac{\|g_k\|^2}{y_{k-1}^T d_{k-1}}$	Dai-Yuan	1999
$\beta_k^{DL} = \dfrac{g_k^T (y_k - t s_k)}{d_k^T y_k}$	Dai-Liao	2001
$\beta_k^{HZ} = \left(y_{k-1} - 2 d_{k-1} \dfrac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}} \right)^T \dfrac{g_k}{d_{k-1}^T y_{k-1}}$	Hager-Zhang	2005
$\beta_k^{*} = \dfrac{g_k^T \left(g_k - \dfrac{\|g_k\|}{\|g_{k-1}\|} g_{k-1} \right)}{\|g_{k-1}\|^2}$	Wei	2006
$\beta_k^{MN} = \dfrac{\|g_k\|^2 - \dfrac{\|g_k\|}{\|g_{k-1}\|} g_k^T g_{k-1}}{\mu \lvert g_k^T d_{k-1} \rvert + \|g_{k-1}\|^2}$	Fan-Zhu-Zhou	2011
$\beta_k^{RMIL} = \dfrac{g_k^T (g_k - g_{k-1})}{\|d_{k-1}\|^2}$	Rivaie et al.	2012

Remark 2.1.3.

- *If $f(x) = \frac{1}{2} x^T Q x - b^T x$ with Q symmetric positive definite, and α_k is obtained using exact line search, then:*

$$\beta_k = \frac{g_k^T Q d_{k-1}}{d_{k-1}^T Q d_{k-1}},$$

and all β_k variants (HS, FR, PRP, CD, LS, DY, HZ, RMIL, ...) coincide. The quadratic conjugate gradient algorithm generates the same sequence $\{x_k\}$.

- *For non-quadratic f, the β_k formulas differ, leading to different behavior in the nonlinear conjugate gradient methods.*

Conjugate gradient algorithm (nonlinear case)

Begin algorithm

- Given an initial point $x_1 \in \mathbb{R}^n$, $\varepsilon > 0$ and $k = 1$.
- Compute $g_1 = \nabla f(x_1)$ and set $d_1 = -g_1$.
- While $\|\nabla f(x_k)\| \geq \varepsilon$ do
  - Compute β_k using a chosen conjugate gradient formula.
  - Set $d_k = -g_k + \beta_k d_{k-1}$.
  - Compute α_k using an exact or inexact line search.
  - Update $x_{k+1} = x_k + \alpha_k d_k$.
  - Compute $g_{k+1} = \nabla f(x_{k+1})$.
  - Set $k = k + 1$.
- End while.

End algorithm.

2.2 Convergence results of the conjugate gradient method

2.2.1 Conditions C1 and C2 and the Zoutendijk theorem

Conditions C1

We say that $f : \mathbb{R}^n \to \mathbb{R}$ satisfies condition C1 if:

1. f is continuously differentiable in a neighborhood $V(\Omega)$, such that

   $\Omega = \{x \in \mathbb{R}^n : f(x) \leq f(x_1); x_1 \in \mathbb{R}^n \text{ initial point}\}$.

2. $\nabla f(x)$ satisfies the Lipschitz condition in $V(\Omega)$, i.e., there exists a constant $L > 0$ such that:

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \text{ for all } x, y \in V(\Omega).$$

Condition C2 (Boundedness condition):

The level set $\Omega = \{x \in \mathbb{R}^n : f(x) \leq f(x_1)\}$ is bounded, i.e., there exists a constant $B < \infty$ such that:

$$\|x\| \leq B \text{ for all } x \in \Omega.$$

Zoutendijk theorem [42]

The Zoutendijk theorem is a key theorem used to prove the convergence of nonlinear conjugate gradient methods. It is especially effective when using inexact line search, such as **Wolfe** and **Goldstein**, and requires some assumptions on the function f, such as differentiability and a Lipschitz condition on its gradient ∇f.

We denote by θ_k the angle formed by the descent direction d_k with $-g_k$,

$$\theta_k = \widehat{d_k, -g_k}.$$

In terms of the inner product, we have:

$$\langle x, y\rangle = \|x\|\|y\| \cos \widehat{(x, y)}.$$

In our case, this gives:

$$\langle -g_k, d_k\rangle = -g_k^T d_k = \|g_k\|\|d_k\| \cos(\theta_k),$$

or equivalently:

$$\cos(\theta_k) = \frac{-g_k^T d_k}{\|g_k\|\|d_k\|}.$$

Theorem 2.2.1 (Zoutendijk)**.** *[2] Let $f : \mathbb{R}^n \to \mathbb{R}$ be a lower bound and differentiable function with a Lipschitz continuous gradient (i.e., f satisfies condition* **C1***).*

Let a sequence $\{x_k\}_{k\in\mathbb{N}}$ in $\mathbb{R}^n$ be generated by the formula (2.1)*.*

Then, we have:

$$\sum_{k\geq 1} \|g_k\|^2 \cos^2\theta_k < \infty. \tag{2.4}$$

2.2.2 Applying the Zoutendijk theorem to show global convergence

The condition $\sum_{k=1}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty$ is equivalent to (2.4) where θ_k is the angle formed by d_k with $-g_k$. It is clear that if

$$\cos(\theta_k) \geq \delta > 0,$$

then we have

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

We are unable to prove this limit, but only:

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{2.5}$$

The condition (2.5) implies that there exists a subsequence of $\{\|g_k\|\}$ that tends to zero.

Sufficient conditions associated to Zoutendijk theorem to show (2.5)

Definition 2.2.2. *We say that d_k is a sufficient descent direction if:*

$$g_k^T d_k \leq -C\|g_k\|^2 \text{ with } C > 0.$$

To demonstrate (2.5), we associate the following two hypotheses with Zoutendijk's theorem:

Hypothesis 1: Sufficient descent direction is assured by d_k, i.e., $g_k^T d_k \leq -C\|g_k\|^2$, with $C > 0$.

Hypothesis 2: There exists a constant $\gamma > 0$ such that:

$$\|d_k\|^2 \leq \gamma k.$$

In summary, we have:

$$\left\{\begin{array}{l} \text{Zoutendijk condition } \left(\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty\right) \\ \textbf{Hypothesis 1} \\ \textbf{Hypothesis 2} \end{array}\right\} \Rightarrow \left\{\liminf_{k\to\infty} \|g_k\| = 0\right\}.$$

2.3 Line search

2.3.1 Line search principle

In addition to the fact that one-dimensional problems are common in applications, line search is essential to all conventional multidimensional optimization techniques.

Performing a line search consists of finding a step-size α_k along a descent direction d_k such that the objective function decreases, i.e., $f(x_k + \alpha_k d_k) \leq f(x_k)$.

Exact line search and inexact or economical line search are the two primary categories of line search techniques.

2.3.2 Exact line search

Since we are attempting to minimize f, it makes sense to try to minimize the criterion along the descent direction d_k and thus determine the step-size α_k as the solution to the one-dimensional problem:

$$\min f(x_k + \alpha d_k),\ \alpha \in \mathbb{R}_+^*. \tag{2.6}$$

For unconstrained multidimensional minimization, we typically use the following scheme. The method chooses a direction of movement d_k which is frequently a descent direction of the objective function f (i.e., $\nabla f(x_k)^T d < 0$). It then takes a step in this direction of the form:

$$x_k \to x_{k+1} = x_k + \alpha_k d_k.$$

Where α_k is the optimal solution to the one-dimensional optimization problem in equation (2.6).

To attain a specific reduction in the target value f, which means that we wish to ensure:

$$f(x_k + \alpha_k d_k) \leq f(x_k + \alpha d_k), \forall \alpha > 0, \tag{2.7}$$

most approaches assume that x_k and d_k are fixed values and that the stepsize α_k is determined by minimizing the one-dimensional function $\varphi_k : \mathbb{R}_+ \to \mathbb{R}$ such that:

$$\alpha \mapsto \varphi_k(\alpha) = f(x_k + \alpha d_k). \tag{2.8}$$

In such a way that,

$$\alpha_k = \alpha^* = \arg\min_{\alpha>0} \varphi_k(\alpha). \tag{2.9}$$

In other words, α_k is the local minimum of f which guarantees the largest decrease of f at the point x_k in the direction d_k.

Remark 2.3.1. *The reason that we say "exact" is when φ_k is quadratic, we obtain the line search solution exactly.*

The advantage of exact line search

The main advantage of the exact line search method is that the obtained step α_k is optimal, i.e., $\alpha_k = \alpha^*$, satisfying:

$$\varphi_k(\alpha_k) = \varphi_k(\alpha^*) = \min\{\varphi_k(\alpha) : \alpha \in (0, +\infty)\}.$$

In other words, if $\alpha_k = \alpha^*$ the function f decreases the most between the points x_k and x_{k+1}.

The inconvenience of exact line search

More generally for some nonlinear function f:

1) The steps of these methods usually involve lengthy calculation time and cannot be taken with finite precision.

2) Many advanced optimization algorithms don't use exact line search because finding the best step-size α^*, often requires minimizing the function φ at each iterations, which can take a lot of time. Instead, we use an inexact line search, where we just look for a step-size α that reduces the function value enough, without needing to be optimal.

2.3.3 Inexact line search

In iteration k, we have the iterate $x_k \in \mathbb{R}^n$ as well as the corresponding search direction $d_k \in \mathbb{R}^n$ that is a descent direction for our objective $f : \mathbb{R}^n \to \mathbb{R}$ such that $\nabla f(x_k)^T d_k < 0$.

The aim is to substantially decrease the value of the objective by a displacement α_k in the direction of d_k. A number of rules have been created for this purpose by many researchers (**Armijo**, **Goldstein**, **Wolfe**, **Albaali**, **Fletcher**,... etc.). In the following, we will present the main rules used.

Armijo rule (1966)

Let $f : \mathbb{R}^n \to \mathbb{R}$, $x \in \mathbb{R}^n$, and $d_k \in \mathbb{R}^n$ a descent direction.

Armijo rule imposes a constraint on the choice of α_k sufficient to locally minimize φ_k.

A natural assumption is to require that f drops by at least a value $\rho \in (0, 1)$ of what the linear approximation of f at x_k predicts. From here, we arrive at the following inequality, sometimes referred to as the Armijo condition or sufficient decrease condition:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \rho\alpha\nabla^T f(x_k) d_k. \tag{2.10}$$

In other words:

$$\varphi_k(\alpha_k) \leq \varphi_k(0) + \rho\varphi_k'(0)\alpha. \tag{2.11}$$

Armijo test

If $\varphi_k(\alpha) \leq \varphi_k(0) + \rho\varphi_k'(0)\alpha$, then α is suitable, else α is too large.

Remark 2.3.2. *In general, in Armijo rule, we ensure that α_k is not too small as otherwise it could harm the convergence of the algorithm, potentially causing it to prematurely converge to a non-stationary point.*

Wolfe rule (1969)

We will present the weak Wolfe conditions with $\alpha > 0$:

$$f(x_k + \alpha d_k) \leq f(x_k) + \rho\alpha\nabla^T f(x_k) d_k. \tag{2.12}$$

$$\nabla^T f(x_k + \alpha d_k) d_k \geq \sigma\nabla^T f(x_k) d_k. \tag{2.13}$$

Where the constants ρ and σ are chosen such that:

$$0 < \rho < \sigma < 1.$$

Wolfe test (weak Wolfe)

If $\varphi_k(\alpha) \leq \varphi_k(0) + \rho\varphi'_k(0)\alpha$ and $\varphi'_k(\alpha) \geq \sigma\varphi'_k(0)$, then α is suitable.

If $\varphi_k(\alpha) > \varphi_k(0) + \rho\varphi'_k(0)\alpha$, then α is too large.

If $\varphi'_k(\alpha) < \sigma\varphi'_k(0)$, then α is too small.

Remark 2.3.3. *The Wolfe rule is more costly than the Armijo rule. However, in many applications, the computation of the gradient $\nabla f(x)$ is at a relatively low additional cost compared with computing $f(x)$, which is why this rule is extensively used.*

Strong Wolfe rule

For some algorithms (e.g., nonlinear conjugate gradient), a stronger condition than equation (2.13) is sometimes needed. It is instead replaced by:

$$|\nabla^T f(x_k + \alpha d_k)d_k| \leq \sigma|\nabla^T f(x_k)d_k| = -\sigma\nabla^T f(x_k)d_k.$$

This gives a rise to the strong Wolfe conditions:

$$f(x_k + \alpha d_k) \leq f(x_k) + \rho\alpha\nabla^T f(x_k)d_k, \tag{2.14}$$

$$|\nabla^T f(x_k + \alpha d_k)d_k| \leq -\sigma\nabla^T f(x_k)d_k, \tag{2.15}$$

where $0 < \rho < \sigma < 1$.

Relaxed Wolfe rule (1996)

Proposed by Dai and Yuan, this rule consists of choosing the step-size satisfying the conditions:

$$f(x_k + \alpha d_k) \leq f(x_k) + \rho\alpha\nabla^T f(x_k)d_k, \tag{2.16}$$

$$\sigma_1\nabla^T f(x_k)d_k \leq \nabla^T f(x_k + \alpha d_k)d_k \leq -\sigma_2\nabla^T f(x_k)d_k, \tag{2.17}$$

where $0 < \rho < \sigma_1 < 1$ and $\sigma_2 > 0$.

Remark 2.3.4. *It's clear that the relaxed Wolfe conditions imply the strong Wolfe conditions and these imply the weak Wolfe conditions.*

Backtracking rule

In elementary form, Backtracking proceeds as follows:

Backtracking line search algorithm

Begin algorithm

- Given a function f, a search direction d_k at point x_k, parameters $0 < \rho < 0.5$ and $\mu \in (0, 1)$, and initial step-size $\alpha = 1$.
- While
$$f(x_k + \alpha d_k) > f(x_k) + \rho\alpha\nabla f(x_k)^T d_k,$$
take $\alpha = \mu\alpha$.
- Endwhile
- Set $\alpha_k = \alpha$.

End algorithm.

Wu rule

Commonly, we compute the step-size α_k using the Wolfe condition. Such a line search is expensive in number of evaluations, and when we prepare for large-scale problems the number of evaluations, either of the function or its derivatives, becomes painfully high.

Let f be a continuously differentiable and strongly convex function with Lipschitz continuous gradient, i.e., there exist constants $\tau > 0$ and $\kappa \geq 0$ such that:

$$[\nabla f(x_{k+1}) - \nabla f(x_k)]^T (x_{k+1} - x_k) \geq \kappa \|x_{k+1} - x_k\|^2, \forall x_{k+1}, x_k \in \mathbb{R}^n, \tag{2.18}$$

and

$$\|\nabla f(x_{k+1}) - \nabla f(x_k)\| \leq \tau \|x_{k+1} - x_k\|, \forall x_{k+1}, x_k \in \mathbb{R}^n. \tag{2.19}$$

In [41], Yao et al. propose this formula (without full line search) for computing the step-size α_k:

$$\alpha_k = \frac{-\delta g_k^T d_k}{\|d_k\|_{Q_k}^2}, \tag{2.20}$$

where $\|d_k\|_{Q_k} = \sqrt{d_k^T Q_k d_k}$, $\delta \in (0, v_{\min}/\tau)$, τ is the Lipschitz constant of ∇f, and $\{Q_k\}$ is a sequence of positive definite matrices satisfying with positive constants $v_{\min}$ and $v_{\max}$, the following inequalities:

$$v_{\min} d^T d \leq d^T Q_k d \leq v_{\max} d^T d, \forall k, \forall d \in \mathbb{R}^n.$$

But the formula (2.20) depends on positive definite matrices, which will be expensive for higher dimensional optimization problems.

Recently, Wu [40] derived the formula for α_k as follows:

$$\alpha_k = \frac{-\delta g_k^T d_k}{(\bar{g}_{k+1} - g_k)^T d_k + \gamma \theta_k}, \tag{2.21}$$

where

$$\theta_k = 6(f_k - \bar{f}_{k+1}) + 3(g_k + \bar{g}_{k+1})^T d_k,$$

and $f_k = f(x_k)$, $g_k = \nabla f(x_k)$, $\bar{f}_{k+1} = f(x_k + d_k)$ and $\bar{g}_{k+1} = \nabla f(x_k + d_k)$.

δ and γ are parameters satisfying:

$$\delta \in (0, \frac{\kappa}{\tau}),$$

and

$$\gamma \geq 0 \text{ if } \tau = \kappa, \text{ or } \gamma \in \left(0, \frac{\kappa - \delta\tau}{3(\tau - \kappa)}\right) \text{ if } \tau > \kappa,$$

where κ and τ satisfy (2.18) and (2.19).

...

In the next two chapters, we present our contributions, which play an important role in improving the speed of convergence and the behavior of the obtained algorithms.

Chapter 3

Efficient descent direction of a conjugate gradient algorithm for nonlinear optimization

We propose an efficient variant of the conjugate gradient method for nonlinear optimization based on a new parameter β_k. We show that the new search direction fills the sufficient descent condition and we prove the global convergence of the corresponding algorithm using the strong Wolfe inexact line search. The established numerical results show that the new algorithm is more efficient in comparison with the standard Fletcher-Reeves method in terms of either the iteration number or CPU time. These results are published in the international journal "Nonlinear Dynamics and Systems Theory" [30].

3.1 Introduction

Consider the following unconstrained nonlinear optimization problem:

$$\begin{cases} \min \ f(x) \\ x \in \mathbb{R}^n \end{cases}, \tag{3.1}$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a continuously differentiable function.

Conjugate gradient methods are efficient to solve unconstrained optimization problem (3.1), especially for large scale problems. These methods and starting by $x^* \in \mathbb{R}^n$ generate

the following sequence:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{3.2}$$

where x_k is the current iterate point, $\alpha_k > 0$ is the step-size which can be found by one of the line search methods, and d_k is the search direction defined by

$$d_k = \begin{cases} -g_k & \text{for } k = 1 \\ -g_k + \beta_k d_{k-1} & \text{for } k \geq 2 \end{cases}, \tag{3.3}$$

where $g_k = \nabla f(x_k)$ is the gradient of f at x_k, and β_k is a scalar conjugacy coefficient.

Different conjugate gradient methods correspond to different values of the coefficient β_k, a survey on these methods was given by Hager an Zhang in [18].

Recently, a great contribution in the area of conjugate gradient methods and their application has been done by Andrei in [2].

Among the well known formulas of β_k, we can cite Hestenes-Stiefel (HS) [19], Fletcher-Reeves (FR) [15], Polak-Ribière-Polyak (PRP) [33, 32], Conjugate Descent-Fletcher (CD) [14], Liu-Story (LS) [26] and Dai-Yuan (DY) [7], which are given as follows:

$$\beta_k^{HS} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}},\ \beta_k^{FR} = \frac{\| g_k \|^2}{\| g_{k-1} \|^2},\ \beta_k^{PRP} = \frac{g_k^T y_{k-1}}{\| g_{k-1} \|^2},$$

$$\beta_k^{CD} = -\frac{\| g_k \|^2}{g_{k-1}^T d_{k-1}},\ \beta_k^{LS} = -\frac{g_k^T y_{k-1}}{g_{k-1}^T d_{k-1}},\ \beta_k^{DY} = \frac{\| g_{k-1} \|^2}{d_{k-1}^T y_{k-1}},$$

where $y_{k-1} = g_k - g_{k-1}$.

If f is a strongly convex quadratic function and the line search is exact, then all the above formulas of β_k are the same.

In the general case where f is non-quadratic, the algorithms corresponding to each β_k have different numerical performances.

There are many other conjugate gradient methods such as those where the scalar is given by parametric formulas like the β_k proposed by Sellami and Chaib in [36, 37].

Several researchers have also proposed hybrid conjugate gradient methods combining the existing β_k [3, 8, 10, 11, 12, 17] .

There are many convergence results with some line search conditions which have been

widely studied, there the method can guarantee the descent property of each direction which provided the step length α computed by carrying out a line search and it satisfies the strong Wolfe conditions such that

$$f(x_k + \alpha d_k) \leq f(x_k) + \rho\alpha g_k^T d_k \tag{3.4}$$

$$\left|g_{k+1}^T d_k\right| \leq -\sigma g_k^T d_k, \tag{3.5}$$

where $0 < \rho < \sigma < \frac{1}{2}$.

The aim of this chapter is to ameliorate the conjugate gradient method for nonlinear optimization using a new parameter which leads to a new descent direction.

3.2 Convergence analysis of the algorithm based on the new parameter β_k

In this section, we propose a new conjugate gradient coefficient using the Fletcher-Reeves formula [15] with a modification in its denominator. This coefficient is defined as follows:

$$\beta_k^{MSD} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2 + \mu \left|g_k^T d_{k-1}\right|}, \tag{3.6}$$

wher MSD refers to the researchers names (Mohamed, Samia and Djamel) and $\mu > 0$.

Recall that the case $\mu = 0$ corresponds to the Fletcher-Reeves coefficient [15].

3.2.1 Description of the conjugate gradient algorithm

Algorithm MSD

Begin algorithm

- Given a starting point $x_1 \in \mathbb{R}^n$ and a parameter $\varepsilon > 0$.
- Set $k = 1$ and compute $d_1 = -g_1$.
- While $\|g_k\| > \varepsilon$ do
- Find $\alpha_k > 0$ satisfying the strong Wolfe conditions $(3.4, 3.5)$.
- Take $x_{k+1} = x_k + \alpha_k d_k$.

- Compute β_{k+1} by the new formula (3.6).
- Set $d_{k+1} = -g_{k+1} + \beta_{k+1} d_k$ and $k = k + 1$.
- End while.

End algorithm.

3.2.2 Sufficient descent property and global convergence analysis

We make the following basic assumptions on the objective function in order to establish the global convergence results for the new algorithm.

Assumptions

- (i) f is a lower bounded function on the level set

$$\Omega = \{x \in \mathbb{R}^n : f(x) \leq f(x_1)\}.$$

- (ii) In some neighborhood Ω_0 of Ω, f is differentiable and its gradient $g(x)$ is Lipschitz continuous, namely, there exists a constant $L > 0$ such that

$$\|g(x) - g(y)\| \leq L\|x - y\|, \forall x, y \in \Omega_0. \tag{3.7}$$

Under these assumptions, there exists a constant $\varepsilon > 0$ such that

$$\|g_k\| \leq c, \forall k. \tag{3.8}$$

To prove the global convergence of the nonlinear conjugate gradient methods, we use the following lemma, called the Zoutendijk condition [42].

Lemma 3.2.1. *[42] Suppose that the assumptions $(i), (ii)$ hold. Let the sequence $\{x_k\}$ be generated by (3.2) and d_k satisfy $g_k^T d_k < 0$. If α_k is determined by the Wolfe line search conditions, then we have*

$$\sum_{k \geq 1} \frac{\left(g_k^T d_k\right)^2}{\|d_k\|^2} < +\infty.$$

Lemma 3.2.2. *Suppose that the assumption (ii) holds, let the sequence $\{x_k\}$ be generated by (3.2) and the step length α_k satisfy the strong Wolfe conditions with $0 < \sigma < \frac{1}{2}$, then for any k,*

$$-\frac{1}{1-\sigma} \leq \frac{g_k^T d_k}{\|g_k\|^2} \leq \frac{2\sigma - 1}{1-\sigma}. \tag{3.9}$$

As soon as $g_k \neq 0$ for all k, the descent property of d_k is satisfied, i.e.„

$$g_k^T d_k < 0. \tag{3.10}$$

Proof: The lemma is proved by induction. For $k = 1$, since $d_1 = -g_1$, the relations (3.9) and (3.10) are true.

For some $k > 1$, we suppose that (3.9) and (3.10) are true.

By using (3.3), we get

$$\frac{g_{k+1}^T d_{k+1}}{\|g_{k+1}\|^2} = -1 + \frac{\beta_{k+1} g_{k+1}^T d_k}{\|g_{k+1}\|^2}. \tag{3.11}$$

From the second strong Wolfe condition (3.5) and (3.11), we get

$$-1 + \sigma \frac{\beta_{k+1} g_k^T d_k}{\|g_{k+1}\|^2} \leq \frac{g_{k+1}^T d_{k+1}}{\|g_{k+1}\|^2} \leq -1 - \sigma \frac{\beta_{k+1} g_k^T d_k}{\|g_{k+1}\|^2}. \tag{3.12}$$

By using (3.6), we have

$$-1 + \sigma \frac{g_k^T d_k}{\|g_k\|^2 + \mu \left| g_{k+1}^T d_k \right|} \leq \frac{g_{k+1}^T d_{k+1}}{\|g_{k+1}\|^2} \leq -1 - \sigma \frac{g_k^T d_k}{\|g_k\|^2 + \mu \left| g_{k+1}^T d_k \right|}. \tag{3.13}$$

Observe that for all k, we have

$$\frac{1}{\|g_k\|^2 + \mu \left| g_{k+1}^T d_k \right|} \leq \frac{1}{\|g_k\|^2}. \tag{3.14}$$

By introducing (3.14) in (3.13), we get

$$-1 + \sigma \frac{g_k^T d_k}{\|g_k\|^2} \leq \frac{g_{k+1}^T d_{k+1}}{\|g_{k+1}\|^2} \leq -1 - \sigma \frac{g_k^T d_k}{\|g_k\|^2}. \tag{3.15}$$

From (3.9), it follows that for all k,

$$-1-\frac{\sigma}{1-\sigma}\leq\frac{g_{k+1}^{T}d_{k+1}}{\|g_{k+1}\|^{2}}\leq-1+\frac{\sigma}{1-\sigma},$$

which implies

$$-\frac{1}{1-\sigma}\leq\frac{g_{k+1}^{T}d_{k+1}}{\|g_{k+1}\|^{2}}\leq\frac{2\sigma-1}{1-\sigma}.$$

This gives the formula (3.9).

Since $0<\sigma<\frac{1}{2}$, it results from (3.9) that $g_k^T d_k<0$.

This completes the proof of the lemma.

Theorem 3.2.3. *Consider the sequence $\{x_k\}$ generated by (3.2), where d_k is defined by (3.3) and satisfies $g_k^T d_k<0$ and suppose that the assumptions (i) and (ii) hold. Then the Algorithm MSD either stops at a stationary point or converges in the sense that*

$$\lim_{k\longrightarrow\infty}\inf\|g_k\|=0. \tag{3.16}$$

Proof: From the second strong Wolfe condition (3.5) and (3.9), we get

$$\left|g_k^T d_{k-1}\right|\leq-\sigma g_{k-1}^T d_{k-1}\leq\frac{\sigma}{1-\sigma}\|g_{k-1}\|^2, \tag{3.17}$$

using (3.3), (3.14) and (3.17), we obtain

$$\begin{aligned}\|d_k\|^2 &= \|g_k\|^2-2\beta_k g_k^T d_{k-1}+\beta_k^2\|d_{k-1}\|^2\\ &\leq \|g_k\|^2+\beta_k\frac{2\sigma}{1-\sigma}\|g_{k-1}\|^2+\beta_k^2\|d_{k-1}\|^2\\ &\leq \|g_k\|^2+\frac{2\sigma}{1-\sigma}\|g_k\|^2+\beta_k^2\|d_{k-1}\|^2\\ &\leq \left(\frac{1+\sigma}{1-\sigma}\right)\|g_k\|^2+\beta_k^2\|d_{k-1}\|^2.\end{aligned} \tag{3.18}$$

After defining $\tau = \frac{1+\sigma}{1-\sigma}$ and applying (3.18) repeatedly, and using (3.14), it follows that

$$\begin{aligned}
\|d_k\|^2 &\leq \tau\|g_k\|^2 + \beta_k^2 \|d_{k-1}\|^2 \\
&\leq \tau \left[\|g_k\|^2 + \beta_k^2 \|g_{k-1}\|^2 + \beta_k^2\beta_{k-1}^2 \|g_{k-2}\|^2 + \cdots + \beta_k^2\beta_{k-1}^2 \cdots \beta_3^2 \|g_2\|^2\right] \\
&\quad +\beta_k^2\beta_{k-1}^2 \cdots \beta_2^2 \|d_1\|^2 \\
&\leq \tau \left[\|g_k\|^2 + \frac{\|g_k\|^4}{\|g_{k-1}\|^2} + \frac{\|g_k\|^4}{\|g_{k-2}\|^2} + \cdots + \frac{\|g_k\|^4}{\|g_2\|^2}\right] + \frac{\|g_k\|^4}{\|g_1\|^2} \\
&\leq \tau\|g_k\|^4 \left[\sum_{i=2}^{k} \frac{1}{\|g_i\|^2}\right] + \frac{\|g_k\|^4}{\|g_1\|^2} \\
&\leq \tau\|g_k\|^4 \left[\sum_{i=1}^{k} \frac{1}{\|g_i\|^2}\right]. \qquad (3.19)
\end{aligned}$$

Now, if (3.16) is not true, then there exists a constant $\varepsilon > 0$ such that $\|g_k\| > \varepsilon$ for all k. From (3.19), we obtain

$$\frac{\|g_k\|^4}{\|d_k\|^2} \geq \frac{\varepsilon^2}{\tau k},$$

which implies

$$\sum_{k=1}^{\infty} \frac{\|g_k\|^4}{\|d_k\|^2} = \infty. \qquad (3.20)$$

By using (3.9), we have

$$\frac{(g_k d_k)^2}{\|d_k\|^2} \geq \left(\frac{1-2\sigma}{1-\sigma}\right)^2 \frac{\|g_k\|^4}{\|d_k\|^2}, \qquad (3.21)$$

which implies

$$\sum_{k=1}^{\infty} \frac{(g_k d_k)^2}{\|d_k\|^2} = \infty, \qquad (3.22)$$

thus contradicting the Zoutendijk condition.

This completes the proof of the theorem.

3.3 Numerical experiments

This section contains two parts.

In the first part, we present 100 numerical experiments using various functions from unconstrained nonlinear optimization problems, selected from [1, 2, 16].

The main objective of these experiments is to evaluate the performance of our proposed algorithm **MSD** (3.6), in comparison with **FR** [15]. We set the tolerance to $\varepsilon = 10^{-6}$, and the stopping criterion is either $\|g_k\| \leq \varepsilon$ or the number of iterations exceeds 4000. The step-size α_k is computed using the strong Wolfe line search with parameters $\sigma = 0.01$ and $\delta = 0.1$. All results were implanted by Matlab R2019 using DELL PC (Intel (R) Core (TM) i7-77000HQ CPU @ 2.80 GHz, 32 Go RAM), on Windows 10.

To determine the best-performing method among the compared algorithms, we consider the following evaluation criteria: number of iterations, CPU time, gradient norm evaluations, and objective function evaluations. To illustrate the overall performance of the proposed method, we employ the performance profile introduced by Dolan and Moré [6]. The benchmark results for this profile are obtained by running a solver $s \in S$ on a set P of n_p test problems, with n_s solvers in total. For each problem p and solver s, the performance is measured by the computational time $t_{p,s}$ required to solve the problem.

To enable comparison, the performance of each solver s on problem p is evaluated against the best performance achieved by any solver on the same problem. The performance ratio is defined as

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}}.$$

Assuming there exists a constant $r_M \geq r_{p,s}$ for all p and s, and $r_{p,s} = r_M$ if and only if solver s fails to solve problem p, we define the performance profile as

$$\rho(\tau) = \frac{1}{n_p}\text{size}\{p \in P : r_{p,s} \leq \tau\},$$

where $\rho(\tau)$ denotes the probability that solver s performs within a factor $\tau \in \mathbb{R}$ of the best possible performance ratio. The function ρ represents the cumulative distribution function of the performance ratio, and the value $\rho(1)$ indicates the probability that a solver is the best among all solvers. In this section, we present some numerical tests on a set of test functions [1, 2] of unconstrained nonlinear optimization problems using Matlab language.

In the tables of results, we designate by

- n: the size of the problem

- *iter*: the number of iterations
- *time*: the total time in seconds required to complete the evaluation process.

Table 3.1: Comparative results of β_k^{MSD} and β_k^{FR}.

Test function	Size	β_k^{MSD}		β_k^{FR}	
	n	*iter*	*time*	*iter*	*time*
Raydan 1	100	8	0,000411	15	0,000535
	200	8	0,000604	15	0,000852
	500	8	0,001085	15	0,001769
	1000	8	0,002022	15	0,003233
Raydan 2	100	16	0,000421	52	0,001469
	200	16	0,000543	57	0,002259
	500	21	0,001113	3241	0,232550
	1000	21	0,001937	NAN	NAN
Diagonal 4	100	41	0,003746	63	0,005513
	200	42	0,007477	60	0,009580
	500	44	0,016702	62	0,020083
	1000	44	0,038645	99	0,081462
Extended Woods	100	369	0,062696	839	0,162356
	200	471	0,131264	782	0,233382
	500	430	0,271904	659	0,472802
	1000	430	0,029804	932	1,597367
HIMMELBC	100	30	0,004411	42	0,004066
	200	30	0,006783	43	0,007018
	500	32	0,014692	43	0,016906
	1000	32	0,054439	44	0,040352
DIXMAANC	100	14	0,005565	32	0,011920
	200	15	0,012137	23	0,016665
	500	18	0,030703	21	0,033251
	1000	16	0,054192	17	0,073482

Remark 3.3.1. *In the results tables, NAN means that the algorithm does not display the optimal solution after a maximum number of iterations* k_{max}.

Test function	Size	β_k^{MSD}		β_k^{FR}	
	n	$iter$	$time$	$iter$	$time$
HARKERP	100	91	0,004411	302	0,014902
	200	98	0,006783	102	0,007951
	500	108	0,014692	156	0,020768
	1000	255	0,054439	NAN	NAN
PROD1	100	26	0,020014	36	0,022772
	200	26	0,075946	37	0,099651
	500	13	0,262461	13	0,274593
	1000	13	1,105908	14	1,160563
Extended Block	100	23	0,002571	103	0,016227
Diagonal 1	200	24	0,004266	107	0,026036
	500	24	0,013149	103	0,064142
	1000	25	0,028290	114	0,144926
Extended Maratos	100	41	0,002943	42	0,005326
	200	41	0,005166	42	0,009220
	500	41	0,012851	42	0,021821
	1000	42	0,029804	43	0,045364
DIXMAANB	100	13	0,005753	19	0,007854
	200	13	0,011356	20	0,014953
	500	12	0,023140	20	0,031973
	1000	12	0,041619	22	0,066770
Extended Beale	100	66	0,033441	79	0,041068
	200	66	0,061550	128	0,136998
	500	70	0,155799	90	0,220902
	1000	70	0,312607	132	0,665937

Test function	Size	β_k^{MSD}		β_k^{FR}	
	n	*iter*	*time*	*iter*	*time*
Extended White	100	48	0,021397	63	0,025315
and holst	200	48	0,037467	74	0,054997
	500	51	0,093806	144	0,293700
	1000	52	0,187531	78	0,279982
Quadratic Diagonal	100	102	0,005432	126	0,008635
Perturbed 1	200	155	0,009727	1073	0,065435
	500	280	0,029289	1332	0,145182
	1000	421	0,078164	1983	0,411533
DIXMAANA	100	12	0,005473	19	0,007968
	200	12	0,010762	21	0,015383
	500	13	0,025386	20	0,031282
	1000	13	0,046094	20	0,064289
DENSCHNA	100	35	0,009824	58	0,016107
	200	35	0,020065	61	0,031975
	500	35	0,041322	59	0,066094
	1000	36	0,081710	64	0,149320
Freudenstein	100	136	0,018940	265	0,038783
and Roth	200	115	0,026829	171	0,038423
	500	119	0,065969	145	0,085392
	1000	111	0,142784	112	0,143777
Nondiag	100	138	0,014461	NAN	NAN
	200	136	0,011197	NAN	NAN
	500	240	0,043510	NAN	NAN
	1000	835	0,243155	NAN	NAN

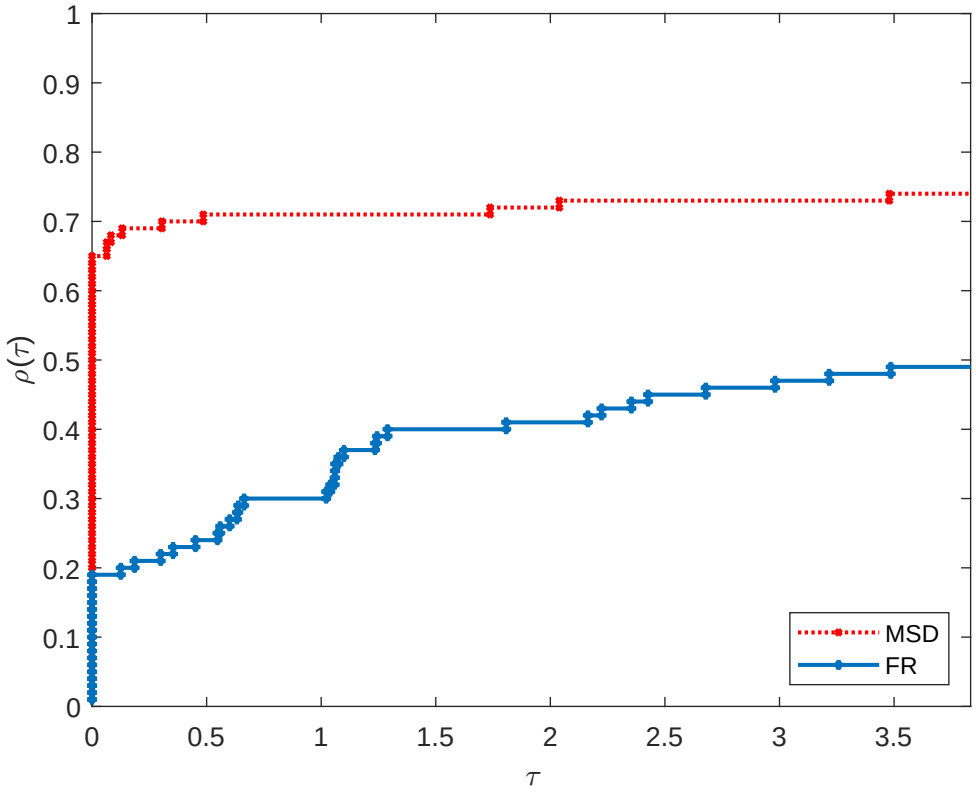

Figure 3.1: Performance profile of MSD and FR based on iteration number.

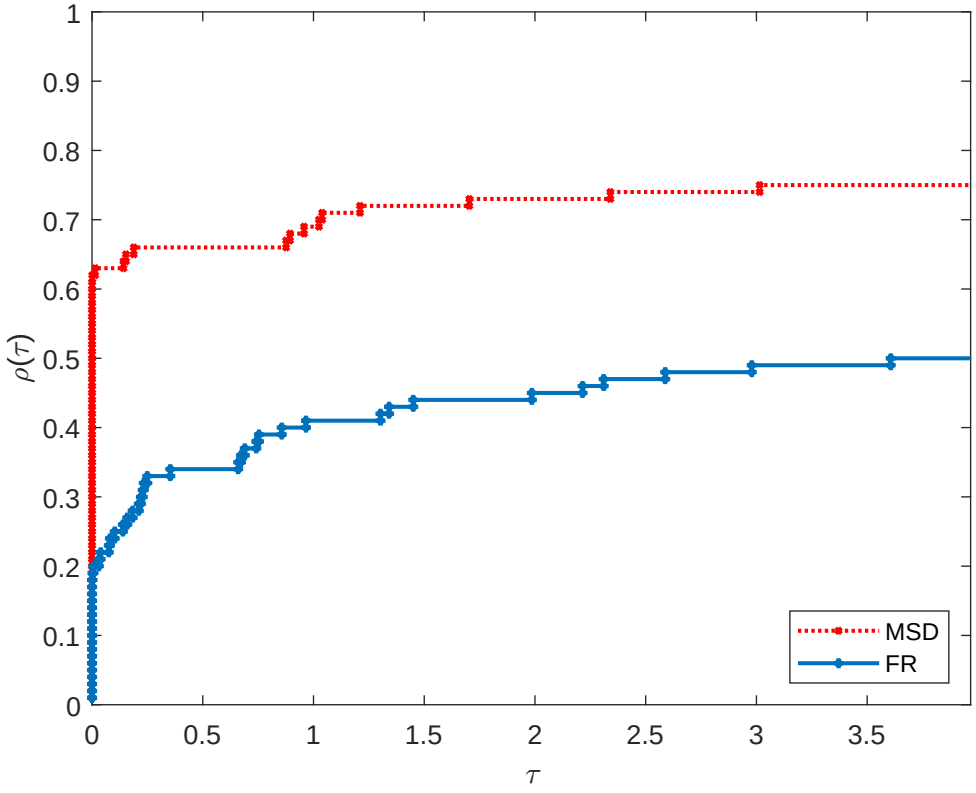

Figure 3.2: Performance profile of MSD and FR based on CPU time.

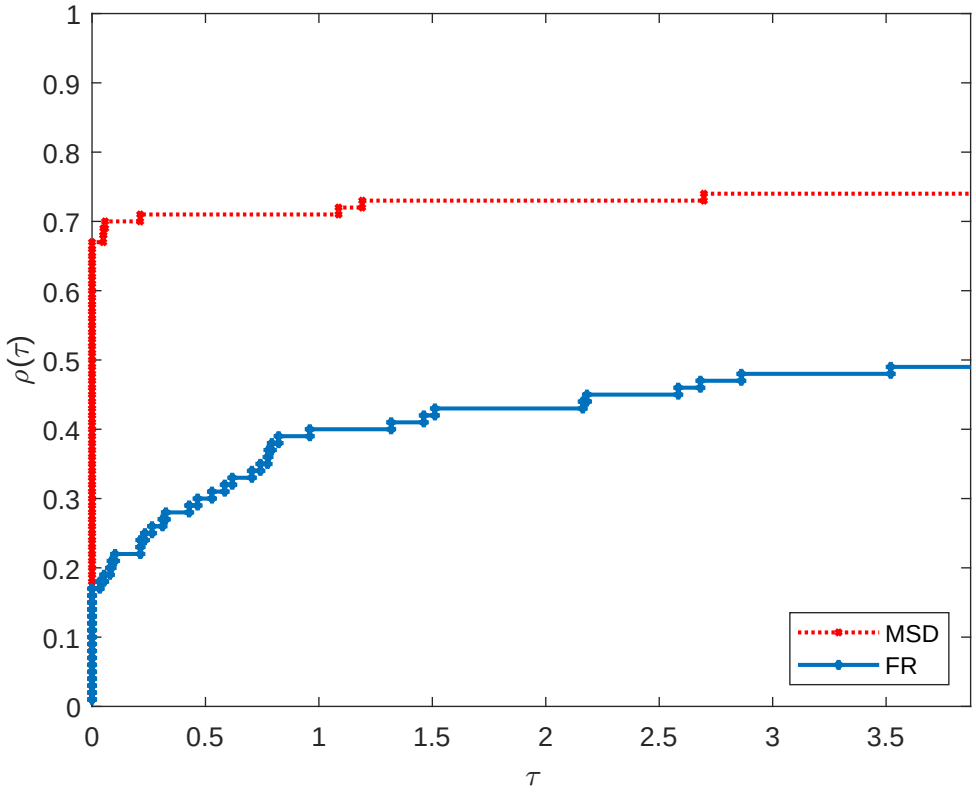

Figure 3.3: Performance profile of MSD and FR based and objective function evaluation.

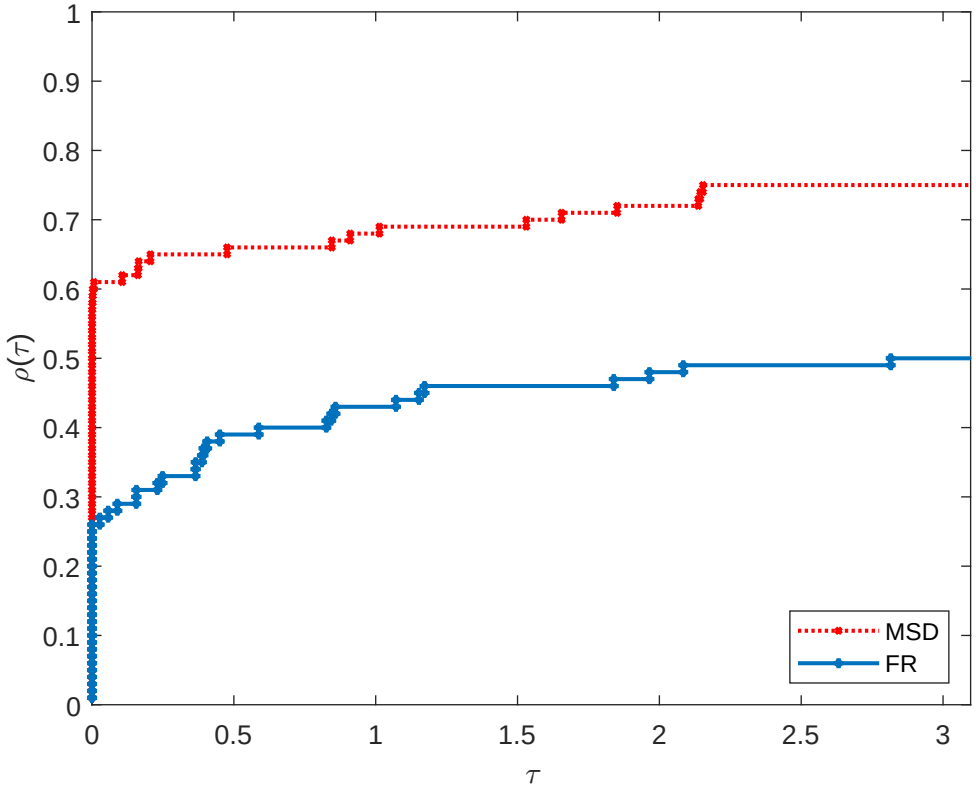

Figure 3.4: Performance profile of MSD and FR based on gradient norm evaluation.

For the second part, we compare the MSD algorithm with FR in image restoration problems involving impulse noise. Here, we use the two-phases scheme considered by Chan et al. [5] to recover an image corrupted by salt-and-pepper noise. In [20], the researchers have used these two phases to make conjugate gradient algorithms capable of restoring images corrupted by impulse noise, even though the noise ratio is high or even reaches 90% .

Let X be an image of size $M \times N$ and $\mathcal{A} = \{1, 2, \ldots, M\} \times \{1, 2, \ldots, N\}$ be the index set of the image X.

$$\min_u F_\alpha(u) = \sum_{(i,j)\in\mathcal{B}} \left(\sum_{(m,n)\in\mathcal{V}_{i,j}\setminus\mathcal{B}} 2\varphi_\alpha(u_{i,j} - y_{m,n}) + \sum_{(m,n)\in\mathcal{V}_{i,j}\cap\mathcal{B}} \varphi_\alpha(u_{i,j} - u_{m,n}) \right),$$

where $\mathcal{B} \subset \mathcal{A}$ is the set of indices of the noise pixels detected in the first phase. $\mathcal{V}_{i,j} = \{(i, j-1), (i, j+1), (i-1, j), (i+1, j)\}$ represents the four closest neighbors for the pixel at location (i, j) for all $(i, j) \in \mathcal{A}$, and $y_{i,j}$ is the observed pixel value at location (i, j). The edge-preserving functional $\varphi_\alpha(t)$ is chosen as $\varphi_\alpha(t) = \sqrt{t^2 + \alpha}$, with $\alpha = 100$ in our tests.

We evaluate the performance using the peak signal-to-noise ratio (PSNR), as

$$PSNR = 10 \log_{10} \frac{255^2}{\frac{1}{MN}\sum_{i,j}(x^r_{i,j} - x^*_{i,j})^2},$$

where $x^r_{i,j}$ and $x^*_{i,j}$ denote the pixel values of the restored and original images, respectively. We test this on Eduard Stiefel (512×512), a Rooster (512×512) and the faculty of science (512×512) images. The stopping criteria is the number of iteration $Iter > 300$ or $\dfrac{|\Delta F_\alpha(u_k)|}{F_\alpha(u_k)} \leq 10^{-4}$.

The noise levels of the salt-and-pepper noise used are as follows: 30%, 50%, 70%, and 90%.

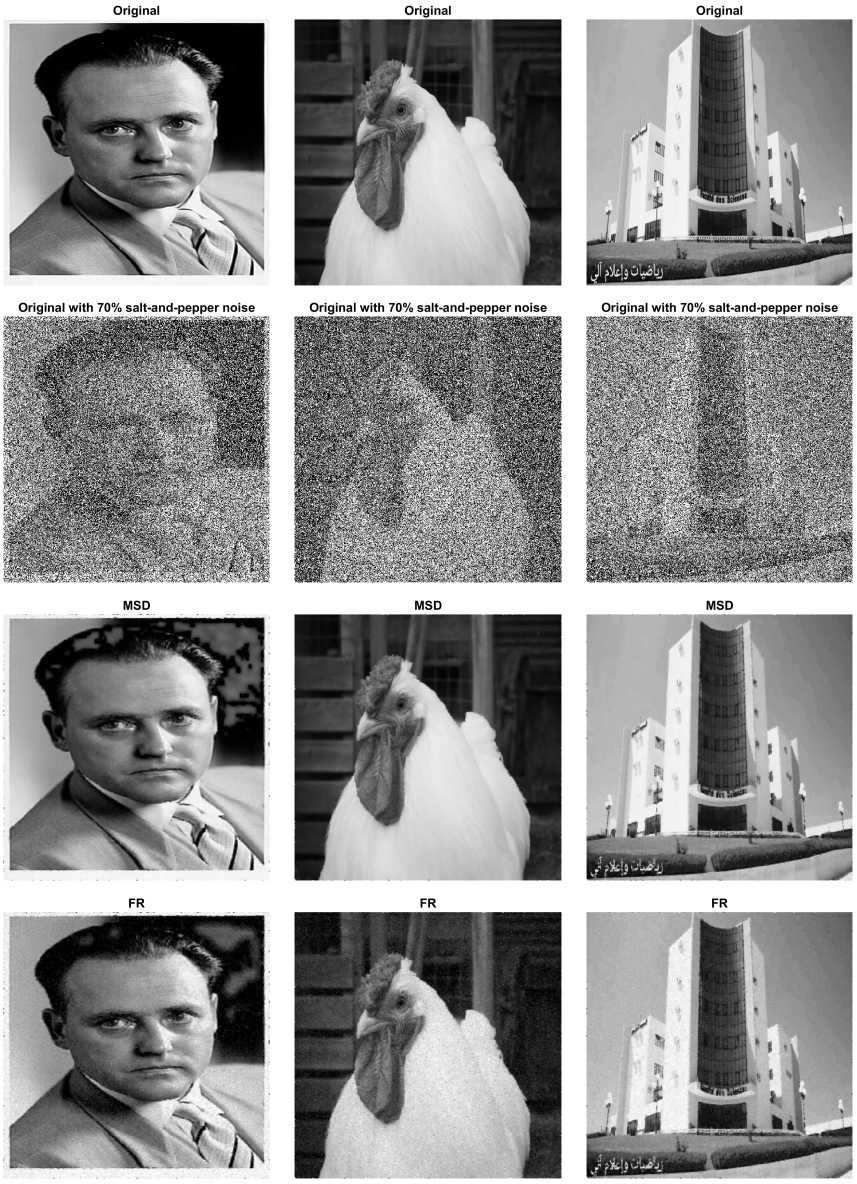

Figure 3.5: The noisy images with 70% salt-and-paper noise and the restored images by MSD and FR.

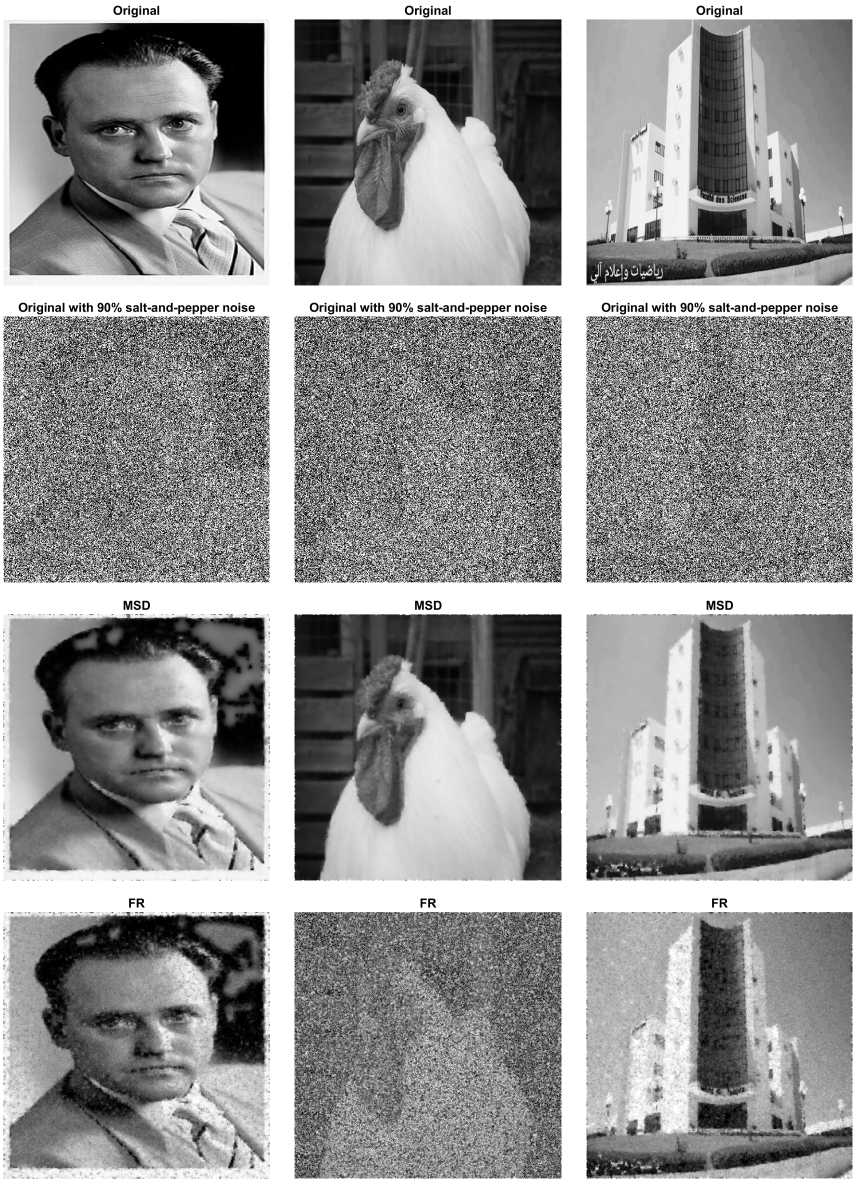

Figure 3.6: The noisy images with 90% salt-and-paper noise and the restored images by MSD and FR.

Image/Noise	MSD			FR		
	It	**CPU (s)**	**PSNR**	**It**	**CPU (s)**	**PSNR**
Stiefel/0.30	24	12.25	29.94	201	77.03	29.63
Stiefel/0.50	31	18.17	26.06	225	131.04	26.35
Stiefel/0.70	22	19.99	21.69	197	187.67	23.09
Stiefel/0.90	39	37.19	19.62	141	155.54	18.93
Rooster/0.30	19	8.22	41.34	8	7.69	21.73
Rooster/0.50	20	15.01	38.14	244	132.68	29.80
Rooster/0.70	20	17.02	34.10	181	135.98	26.09
Rooster/0.90	31	30.34	28.70	5	23.68	9.38
Faculty/0.30	17	7.98	34.78	161	49.50	30.53
Faculty/0.50	18	11.91	31.82	132	75.95	25.93
Faculty/0.70	20	16.73	28.46	239	178.39	26.95
Faculty/0.90	30	33.53	24.19	119	130.19	21.65

Table 3.2: Performance comparison between MSD and FR methods for different images and compression levels.

3.3.1 Commentaries

From the results obtained in the tables above and the performance profile results, as depicted in Figures (3.1), (3.2), (3.3) and (3.4), it is clear that our new algorithm based on the β_k^{MSD} parameter is more efficient than the FR method in terms of the number of iterations and computation time. There is a significant reduction in the number of iterations when using the algorithm MSD compared to FR. On the other hand, when the size of certain examples becomes large $n > 1000$, the FR algorithms fail to provide the optimal solution after the number of iterations $k_{max} = 50000$.

As for image restoration, the MSD method gives an optimal balance between computing time and image quality, particularly in scenarios with high levels of noise. It achieves superior PSNR values and requiring fewer iterations than FR approach.

3.4 Conclusion

We proposed a new β_k, and also provided the proof of the global convergence of the new *Algorithm MSD*. We have proven the effectiveness of this algorithm based on the new β_k^{MSD}. It has a good performance compared with other conjugate gradient methods such as the FR method when considering the selected list of test functions and image restoration problems.

Chapter 4

New parameter of conjugate gradient method for unconstrained nonlinear optimization

We are interested in the performance of nonlinear conjugate gradient methods for unconstrained optimization. In particular, we address the conjugate gradient algorithm with strong Wolfe inexact line search. Firstly, we study the descent property of the search direction of the considered conjugate gradient algorithm based on a new direction obtained from a new parameter. The main objective of this parameter is to improve the speed of convergence of the obtained algorithm. Then, we present a complete study that shows the global convergence of this algorithm. Finally, we establish comparative numerical experiments on well-known test examples to show the efficiency and robustness of our algorithm compared to the algorithm of Hager and Zhang. The set of the results of this work was published in the journal "Statistics, Optimization and Information Computing" [31].

4.1 Introduction

Consider the following unconstrained nonlinear optimization problem

$$\begin{cases} \min \ f(x) \\ x \in \mathbb{R}^n \end{cases}, \tag{4.1}$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a continuously differentiable function.

The nonlinear conjugate gradient methods are efficient to solve problem (4.1), especially for large scale problems, which have the following form:

$$x_1 \in \mathbb{R}^n,\ x_{k+1} = x_k + \alpha_k d_k, \tag{4.2}$$

where x_k is the current iterate point, $\alpha_k > 0$ is the step-size which can be found by one of the line search methods, and d_k is the search direction defined by:

$$d_k = \begin{cases} -g_1 & \text{for } k = 1 \\ -g_k + \beta_k d_{k-1} & \text{for } k \geq 2, \end{cases} \tag{4.3}$$

where $g_k = \nabla f(x_k)$ is the gradient of f at x_k and β_k is a scalar conjugacy coefficient.

The first conjugate gradient method was proposed by Hestenes and Stiefel (HS) in 1952 [19], to solve a linear system of equations. After the introduction of the nonlinear conjugate gradient method of Fletcher and Reeves (FR) in 1964 [15], many parameters β_k have been proposed which give a different conjugate gradient directions d_k. The most famous parameters β_k are those of Polak-Ribiere-Polyak (PRP) [33, 32], Conjugate Descent (CD) [14], Liu-Storey (LS) [26], Dai-Yuan (DY) [7] and Hager-Zhang (HZ) [18], the MN method proposed by Fan et al. in [13] and Rivaie-Mustafa-Ismail-Leong (RMIL) [35]. Later, many combinations and new families of conjugate gradient methods were proposed, like those of Sellami and Chaib [36], the different hybridization methods proposed by Mtagulwa and Kaelo [27] and Dalladji et al. [10].

We cite some formulas of the β_k mentioned above:

$$\beta_k^{HS} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, \quad \beta_k^{FR} = \frac{\| g_k \|^2}{\| g_{k-1} \|^2}, \quad \beta_k^{PRP} = \frac{g_k^T y_{k-1}}{\| g_{k-1} \|^2}, \ \beta_k^{CD} = -\frac{\| g_k \|^2}{g_{k-1}^T d_{k-1}},$$

$$\beta_k^{LS} = -\frac{g_k^T y_{k-1}}{g_{k-1}^T d_{k-1}}, \quad \beta_k^{DY} = \frac{\| g_k \|^2}{d_{k-1}^T y_{k-1}}, \ \beta_k^{RMIL} = \frac{g_k^T y_{k-1}}{\| d_{k-1} \|^2},$$

$$\beta_k^{HZ} = (y_{k-1} - 2d_{k-1}\frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}})\frac{g_k}{d_{k-1}^T y_{k-1}}$$

$$\beta_k^{MN} = \frac{\|g_k\|^2 - \frac{\|g_k\|}{\|g_{k-1}\|} g_k^T g_{k-1}}{\mu\,|g_k^T d_{k-1}| + \|g_{k-1}\|^2}/ \quad \mu > 1,$$

$$\beta_k^{SC} = \frac{(1-\lambda)\,\|g_k\|^2 + \lambda(-g_k^T d_{k-1})}{-g_{k-1}^T d_{k-1}},\ 0<\lambda<1,$$

$$\beta_k^{SC^*} = \frac{\lambda_k\,\|g_k\|^2 + (1-\lambda_k) g_k^T y_{k-1})}{(1-\lambda_k-\mu_k)\,\|g_{k-1}\|^2 + (\lambda_k+\mu_k)(y_{k-1}^T d_{k-1})},$$

$$\text{with } 0<\lambda_k<1 \text{ and } 0<\mu_k<1-\lambda_k,$$

where $y_{k-1} = g_k - g_{k-1}$, $s_{k-1} = x_k - x_{k-1}$ and $\|\ .\ \|$ denotes the Euclidean vector norm. The step-size α_k is determined using the following strong Wolfe line search conditions

$$f(x_k+\alpha_k d_k) \leq f(x_k) + \rho\alpha_k g_k^T d_k \tag{4.4}$$

$$|g_{k+1}^T d_k| \leq -\sigma g_k^T d_k, \tag{4.5}$$

where $0<\rho<\sigma<\frac{1}{2}$.

The aim of this chapter is to propose an efficient conjugate gradient method for non-linear optimization using a new parameter β_k which leads to a new descent direction.

4.2 New formula of β_k and description of the corresponding algorithm

The main ingredients in the conjugate gradient method, which play a very important role in convergence analysis and the behavior of the associated algorithm, are the parameter β_k and the displacement step α_k. In this study, motivated by the recent works of [18] and [39], we present a new conjugate parameter β_k^{OKB} defined as follows:

$$\beta_k^{OKB} = \frac{\|g_k\|^2 - \frac{\|g_k\|}{\|d_{k-1}\|}\left|g_k^T d_{k-1}\right|}{d_{k-1}^T y_{k-1}}, \tag{4.6}$$

where OKB refers to the researchers family names (Ouaoua, Khelladi and Benterki).

4.2.1 Description of the algorithm OKB

Begin algorithm

- Given a starting point $x_1 \in \mathbb{R}^n$ and a parameter $\varepsilon > 0$.
- Set $k = 1$ and compute $d_1 = -g_1$.

- While $\|g_k\| > \varepsilon$ do
  - Find $\alpha_k > 0$ satisfying the strong Wolfe conditions (4.4) and (4.5), $\sigma = 0,4$ and $\rho = 10^{-4}$.
  - Take $x_{k+1} = x_k + \alpha_k d_k$.
  - Compute β_{k+1} by the new formula (4.6).
  - Compute $d_{k+1} = -g_{k+1} + \beta_{k+1} d_k$.
  - Set $k = k + 1$.
- End while.

End algorithm.

4.2.2 Sufficient descent property and global convergence analysis

We start with the following basic assumptions on the objective function in order to establish the global convergence results for the new algorithm.

Assumptions

- (A1) f is bounded below on the level set $\Omega = \{x \in \mathbb{R}^n : f(x) \leq f(x_1)\}$.
- (A2) In some neighborhood Ω_0 of Ω, f is differentiable and it is gradient $g(x)$ is Lipschitz continuous, namely, there exist a constant $L > 0$ such that

$$\|g(x) - g(y)\| \leq L\|x - y\|, \forall x, y \in \Omega_0. \tag{4.7}$$

Under these assumptions, there exist a constant $c > 0$ such that

$$\|g_k\| \leq c, \forall k \geq 1. \tag{4.8}$$

Lemma 4.2.1. *Suppose that Assumption* (*A*2) *holds, let the sequence* $\{x_k\}$ *generated by* (4.2) *and the step-size* α_k *satisfies strong Wolfe conditions* (4.4) *and* (4.5)*, then*

$$g_k^T d_k \leq \frac{2\sigma - 1}{1 - \sigma}\|g_k\|^2. \tag{4.9}$$

Furthermore, for any k as soon as $g_k \neq 0$, the descent property of d_k is satisfied, i.e.,

$$g_k^T d_k < 0. \tag{4.10}$$

Proof: The lemma is proved by induction. For $k = 1$, since $d_1 = -g_1$, therefore (4.9) and (4.10) are verified.

For some $k > 1$, we suppose that (4.9) and (4.10) are true.

By using (4.3), we get

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + \beta_{k+1}^{OKB} g_{k+1}^T d_k, \tag{4.11}$$

and from (4.6), we have

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + \frac{\|g_{k+1}\|^2 - \frac{\|g_{k+1}\|}{\|d_k\|}\left|g_{k+1}^T d_k\right|}{d_k^T y_k} g_{k+1}^T d_k. \tag{4.12}$$

Set $y_k = g_{k+1} - g_k$. Then, from (4.5) we obtain

$$d_k^T y_k = d_k^T g_{k+1} - d_k^T g_k > \sigma d_k^T g_k - d_k^T g_k = (\sigma - 1) d_k^T g_k > 0. \tag{4.13}$$

By substiting (4.5) and (4.13) into (4.12), we obtain

$$g_{k+1}^T d_{k+1} \leq -\|g_{k+1}\|^2 + \frac{\|g_{k+1}\|^2 - \frac{\|g_{k+1}\|}{\|d_k\|}\left|g_{k+1}^T d_k\right|}{(\sigma - 1) d_k^T g_k}(-\sigma d_k^T g_k). \tag{4.14}$$

We know that

$$0 \leq \left|g_{k+1}^T d_k\right| \leq \|g_{k+1}\|.\|d_k\|, \tag{4.15}$$

this leads to

$$0 \leq \|g_{k+1}\|^2 - \frac{\|g_{k+1}\|}{\|d_k\|}\left|g_{k+1}^T d_k\right| \leq \|g_{k+1}\|^2, \tag{4.16}$$

hence, (4.14) becomes

$$-\|g_{k+1}\|^2 \leq g_{k+1}^T d_{k+1} \leq -\|g_{k+1}\|^2 + \frac{\sigma}{1-\sigma}\|g_{k+1}\|^2. \tag{4.17}$$

Now, observe that for all $k \geq 1$, we have

$$-\|g_{k+1}\|^2 \leq g_{k+1}^T d_{k+1} \leq \frac{2\sigma - 1}{1 - \sigma}\|g_{k+1}\|^2. \tag{4.18}$$

Since $0 < \sigma < \frac{1}{2}$, it results that

$$-1 < \frac{2\sigma - 1}{1 - \sigma} < 0. \tag{4.19}$$

Therefore, from (4.19) it results that $g_{k+1}^T d_{k+1} < 0$. This completes the proof.

Lemma 4.2.2. *Suppose that Assumptions* $(A1)$ *and* $(A2)$ *hold. Consider common iterate by* (4.2) *, with* d_k *is a sufficient descent direction and* α_k *is determinate by the strong Wolfe line search condition* (4.4) *and* (4.5)*. Then, the Zoutendjik condition*

$$\sum_{k=1}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty, \tag{4.20}$$

holds.

Proof: See [42].

Theorem 4.2.3. *Consider the iteration* $x_{k+1} = x_k + \alpha_k d_k$*, where* d_k *is defined by* (4.3) *and suppose that Assumption* $(A2)$ *holds, then the new algorithm OKB either stops at stationary point or converges in the sense*

$$\lim_{k \longrightarrow \infty} \inf \|g_k\| = 0. \tag{4.21}$$

Proof: We consider for all the demonstration that $\beta_k = \beta_k^{OKB}$.

We suppose that $\lim\limits_{k \longrightarrow \infty} \inf \|g_k\| \neq 0$ .i.e., $\forall k$, $\exists\ c > 0$ such that

$$\|g_k\| > c. \tag{4.22}$$

From (4.3), we get for all $k \geq 2$

$$\begin{aligned} d_k + g_k &= \beta_k d_{k-1} \\ \|d_k\|^2 + \|g_k\|^2 + 2g_k^T d_k &= \beta_k^2 \|d_{k-1}\|^2 \\ \|d_k\|^2 &= -\|g_k\|^2 - 2g_k^T d_k + \beta_k^2 \|d_{k-1}\|^2 . \end{aligned} \tag{4.23}$$

Dividing by $(g_k^T d_k)^2$, we get

$$\begin{aligned}\frac{\|d_k\|^2}{(g_k^T d_k)^2} &= -\frac{\|g_k\|^2}{(g_k^T d_k)^2} - \frac{2}{g_k^T d_k} + \beta_k^2 \frac{\|d_{k-1}\|^2}{(g_k^T d_k)^2} \\ &= -\left(\frac{\|g_k\|}{g_k^T d_k} + \frac{1}{\|g_k\|}\right)^2 + \frac{1}{\|g_k\|^2} + \beta_k^2 \frac{\|d_{k-1}\|^2}{(g_k^T d_k)^2} \\ &\leq \frac{1}{\|g_k\|^2} + \frac{\|d_{k-1}\|^2}{(g_{k-1}^T d_{k-1})^2} \times \frac{\beta_k^2 (g_{k-1}^T d_{k-1})^2}{(g_k^T d_k)^2}. \end{aligned} \tag{4.24}$$

We remark that $\frac{\beta_k^2 (g_{k-1}^T d_{k-1})^2}{(g_k^T d_k)^2} < 1$.

Indeed, we have

$$\beta_k^2 (g_{k-1}^T d_{k-1})^2 - (g_k^T d_k)^2 = \left(\beta_k g_{k-1}^T d_{k-1} - g_k^T d_k\right)\left(\beta_k g_{k-1}^T d_{k-1} + g_k^T d_k\right),$$

from (4.13) and (4.16), we can deduce that β_k is a postive scalar.

from (4.10), and β_k is a postive scalar, we get

$$\beta_k g_{k-1}^T d_{k-1} + g_k^T d_k < 0, \tag{4.25}$$

and

$$\begin{aligned}\beta_k g_{k-1}^T d_{k-1} - g_k^T d_k &= \beta_k g_{k-1}^T d_{k-1} - g_k^T(-g_k + \beta_k d_{k-1}) \\ &= -\beta_k y_{k-1}^T d_{k-1} + \|g_k\|^2 \\ &= -\|g_k\|^2 + \frac{\|g_k\|}{\|d_{k-1}\|} |g_k d_{k-1}| + \|g_k\|^2 \\ &= \frac{\|g_k\|}{\|d_{k-1}\|} |g_k d_{k-1}| > 0. \end{aligned} \tag{4.26}$$

Then, from (4.25) and (4.26) we get

$$\beta_k^2 (g_{k-1}^T d_{k-1})^2 - (g_k^T d_k)^2 < 0.$$

So, the formula (4.24) becomes

$$\frac{\|d_k\|^2}{(g_k^T d_k)^2} \leq \frac{1}{\|g_k\|^2} + \frac{\|d_{k-1}\|^2}{(g_{k-1}^T d_{k-1})^2}$$

which gives

$$\frac{\|d_k\|^2}{(g_k^T d_k)^2} - \frac{\|d_{k-1}\|^2}{(g_{k-1}^T d_{k-1})^2} \leq \frac{1}{\|g_k\|^2},$$

which implies

$$\frac{\|d_k\|^2}{(g_k^T d_k)^2} - \frac{\|d_1\|^2}{(g_1^T d_1)^2} \leq \sum_{i=2}^{k} \frac{1}{\|g_i\|^2}.$$

As, $d_1 = -g_1$, we have

$$\frac{\|d_1\|^2}{(g_1^T d_1)^2} = \frac{1}{\|g_1\|^2},$$

therefore

$$\frac{\|d_k\|^2}{(g_k^T d_k)^2} - \frac{1}{\|g_1\|^2} \leq \sum_{i=2}^{k} \frac{1}{\|g_i\|^2},$$

which gives

$$\frac{\|d_k\|^2}{(g_k^T d_k)^2} \leq \sum_{i=1}^{k} \frac{1}{\|g_i\|^2}, \tag{4.27}$$

from (4.22) and (4.27), we obtain

$$\frac{(g_k^T d_k)^2}{\|d_k\|^2} > \frac{c^2}{k}.$$

Which implies

$$\sum_{k=1}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} = \infty,$$

thus, contradicting the Zoutendijk condition (4.20) and guarantying (4.21),
i.e., $\lim_{k \longrightarrow \infty} \inf \|g_k\| = 0$. This completes the proof.

4.3 Numerical experiments

The same as in chapter 3, this section parted in two parts. In the first part, we present some numerical experiments on a set of test functions as considered in chapter 3 of unconstrained nonlinear optimization problems using Matlab R2019 and our tests are conducted on DELL PC (Intel (R) Core (TM) i7-77000HQ CPU @ 2.80 GHz, 32 Go RAM), on Windows 10.

The object of this experiments is to show the performance of our new coefficient in comparing with other class of classical existing coefficient. In numerical tests, we consider the **algorithm OKB** based on our new coefficient β_k^{OKB} (4.6) compared with the Hager-

Zhang (HZ) method using β_k^{HZ} [18].

In the table of results we designate by:

- n: the dimension of the problem,
- $iter$: the number of iterations,
- $Time$: the total time in second required to complete the evaluation process.

Table 4.1: Comparative performance of β_k^{OKB} vs β_k^{HZ}.

Test function	Size	β_k^{OKB}		β_k^{HZ}	
	n	$iter$	$time$	$iter$	$time$
DENCHNA	100	30	0,011245	49	0,019950
	200	30	0,016902	50	0,030366
	500	30	0,032006	50	0,075188
	1000	31	0,062500	51	0,131070
Extended Block Diagonal 1	100	25	0,002885	37	0,004162
	200	25	0,004796	37	0,007083
	500	25	0,034679	38	0,047336
	1000	26	0,038550	38	0,063640
DIXMAANA	100	16	0,006063	19	0,007706
	200	16	0,010358	16	0,012170
	500	17	0,034679	18	0,047336
	1000	17	0,064801	18	0,086842
DIXMAANB	100	14	0,005521	17	0,006647
	200	14	0,009459	20	0,013884
	500	14	0,028822	20	0,050333
	1000	15	0,509552	21	0,098544

Test function	Size	β_k^{OKB}		β_k^{HZ}	
	n	*iter*	*time*	*iter*	*time*
Extended Heibert	100	186	0,021666	782	0,097860
	200	262	0,050333	703	0,142559
	500	150	0,149345	682	0,752066
	1000	302	0,325097	836	1,539170
Extended Rosenbrock	100	127	0,015588	361	0,045206
	200	108	0,031787	700	0,149684
	500	199	0,097230	505	0,487145
	1000	121	0,144109	623	0,781642
DIXMAAND	100	20	0,006672	20	0,008616
	200	21	0,013145	22	0,015209
	500	21	0,048954	22	0,056010
	1000	21	0,082428	21	0,103713
Extended White and Holst	100	69	0,023580	97	0,038436
	200	69	0,044379	107	0,083291
	500	84	0,124510	95	0,173380
	1000	72	0,216234	104	0,373296
Extended Beale	100	109	0,046908	267	0,124510
	200	119	0,092719	247	0,222006
	500	127	0,248663	190	0,437382
	1000	115	0,440869	239	1,063390
Extended Woods	100	219	0,088517	944	0,102763
	200	363	0,057954	1007	0,168555
	500	346	0,133696	1072	0,505063
	1000	286	0,280782	1114	1,169708
Raydan 1	100	12	0,000175	20	0,000353
	200	12	0,000311	20	0,000444
	500	12	0,000438	20	0,000901
	1000	16	0,001061	Failed	Failed

Test function	Size	β_k^{OKB}		β_k^{HZ}	
	n	*iter*	*time*	*iter*	*time*
Extended Powell	100	433	0,208014	Failed	Failed
	200	402	0,364384	Failed	Failed
	500	497	1,099740	Failed	Failed
	1000	615	2,729780	Failed	Failed
DIXMAANC	100	16	0,007667	19	0,008081
	200	17	0,012793	18	0,014042
	500	18	0,035370	19	0,049259
	1000	18	0,074232	20	0,095920
Power	100	1140	0,064120	1814	0,138626
	200	2545	0,201966	3487	0,369626
	500	6544	0,830418	9740	1,672919
	1000	10804	2,824616	18669	5,641824
Quadratic Diagonal Perturbed	100	159	0,016168	220	0,028212
	200	200	0,052648	426	0,079868
	500	292	0,026646	384	0,067986
	1000	336	0,081887	548	0,154064
Rodenstein and Roth	100	82	0,016210	169	0,037357
	200	82	0,092719	265	0,222006
	500	69	0,055907	187	0,238901
	1000	96	0,176877	118	0,290437
HARKERP	100	299	0,017223	510	0,029777
	200	383	0,057954	952	0,168555
	500	141	0,027043	243	0,098434
	1000	219	0,059078	362	0,165848
Extended Tridiagonal 1	100	54	0,019141	72	0,029115
	200	54	0,033980	Failed	Failed
	500	54	0,083032	Failed	Failed
	1000	65	0,198614	Failed	Failed

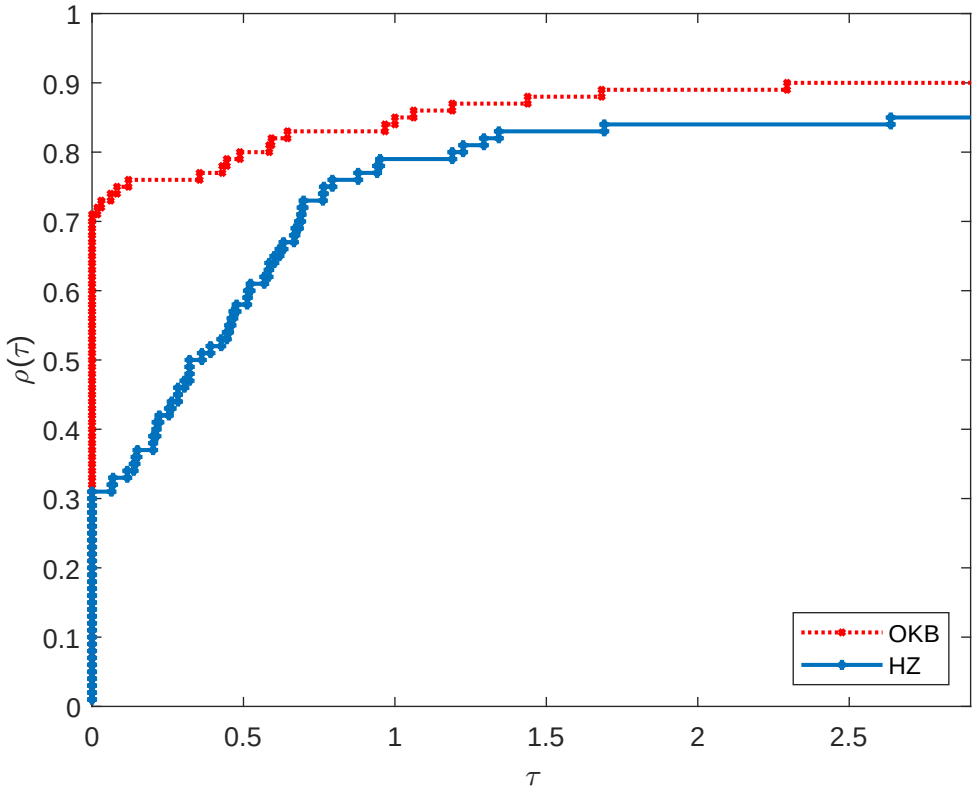

Figure 4.1: Performance profile of OKB and HZ based on iteration number.

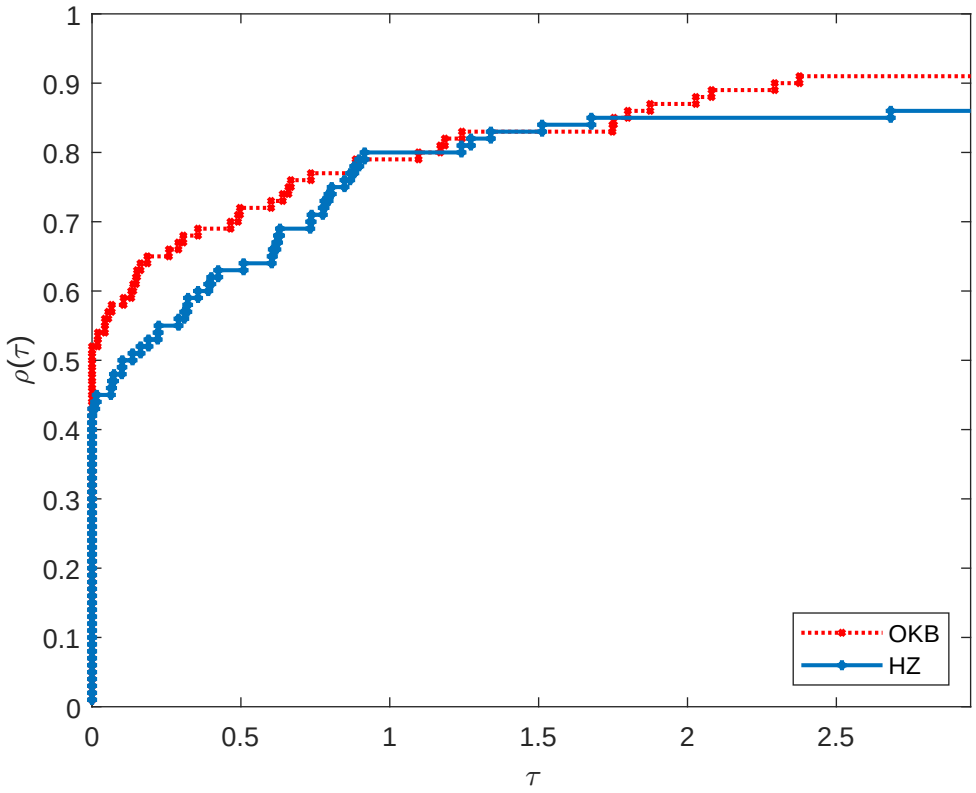

Figure 4.2: Performance profile of OKB and HZ based on CPU time.

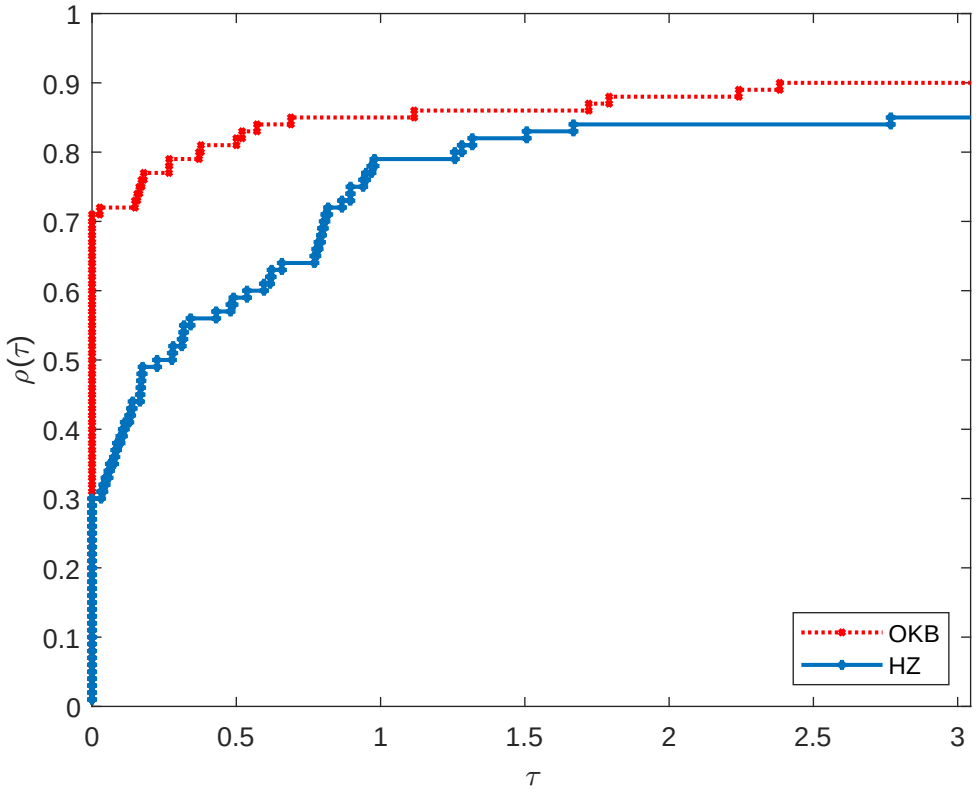

Figure 4.3: Performance profile of OKB and HZ based and objective function evaluation.

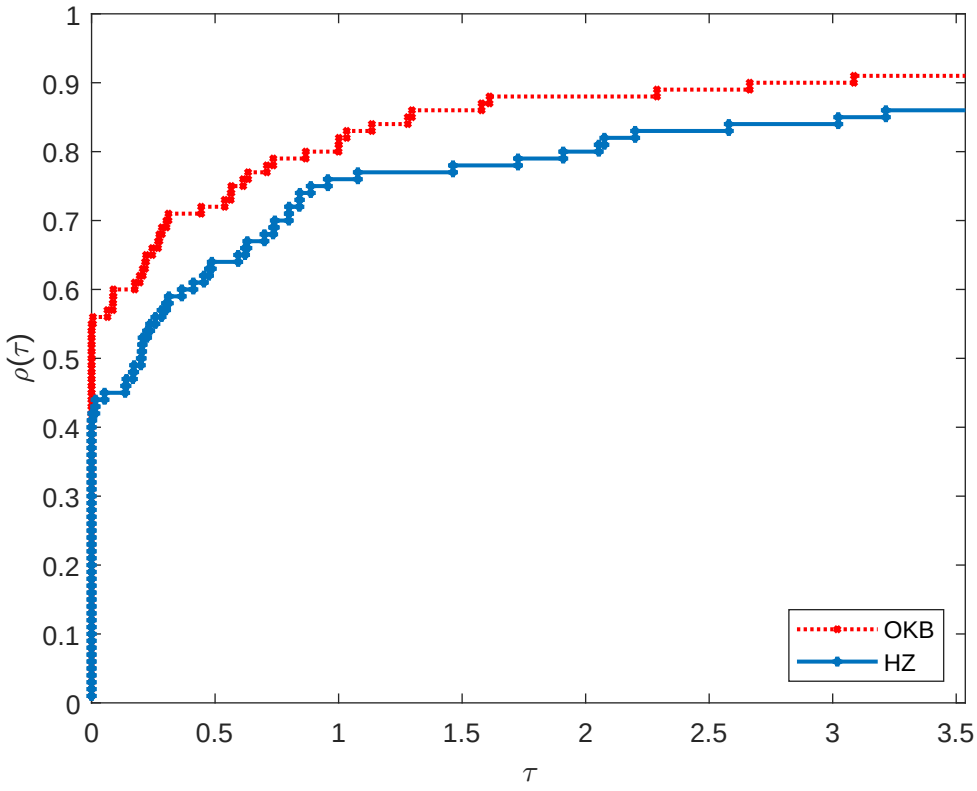

Figure 4.4: Performance profile of OKB and HZ based on gradient norm evaluation.

As for the second part, we implemented our algorithm in application to image restoration problem same as in chapter 3.

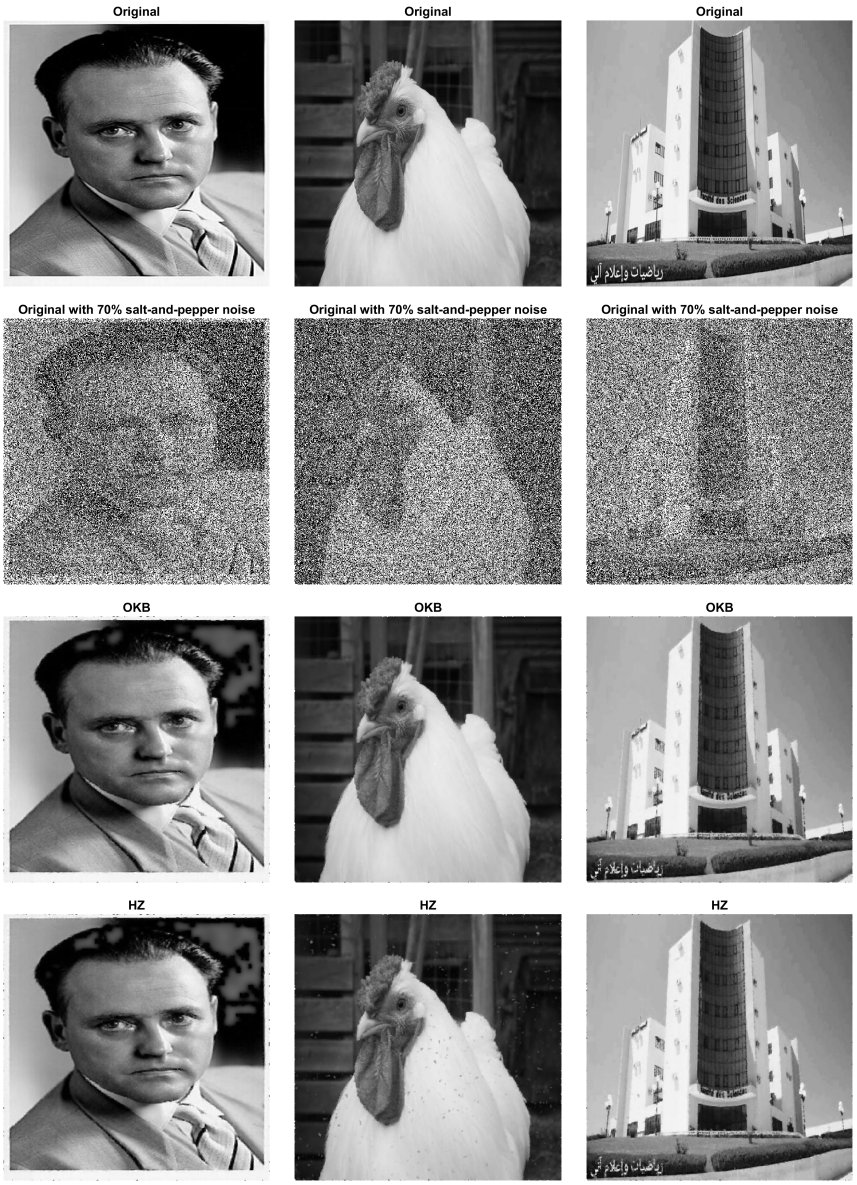

Figure 4.5: The noisy images with 70% salt-and-paper noise and the restored images by OKB and HZ.

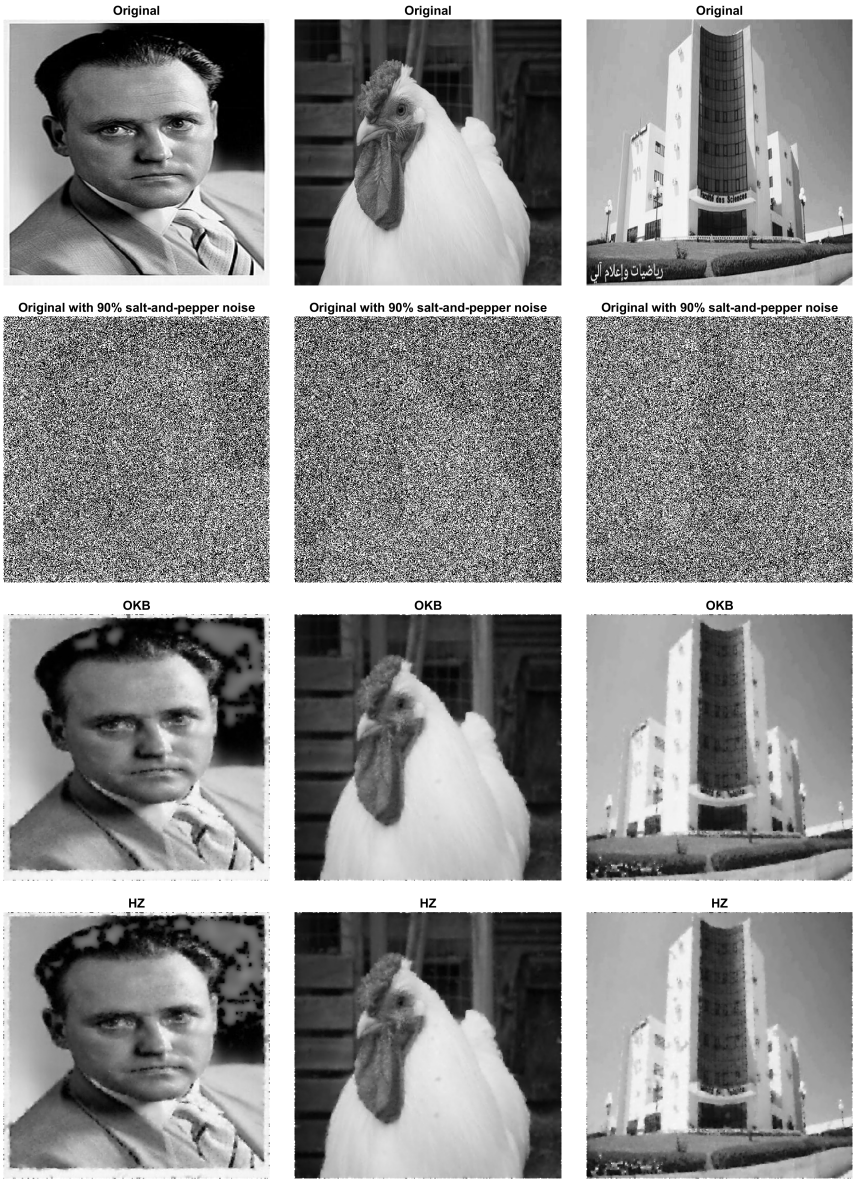

Figure 4.6: The noisy images with 90% salt-and-paper noise and the restored images by OKB and HZ.

Image/Noise	OKB			HZ		
	It	**CPU (s)**	**PSNR**	**It**	**CPU (s)**	**PSNR**
Stiefel/0.30	25	12.26	30.38	30	13.90	29.21
Stiefel/0.50	32	21.47	26.23	39	25.83	25.35
Stiefel/0.70	41	31.68	23.77	43	36.07	22.65
Stiefel/0.90	44	47.01	19.88	48	45.25	18.45
Rooster/0.30	14	9.27	41.27	22	8.72	41.12
Rooster/0.50	19	16.28	38.13	25	27.63	37.99
Rooster/0.70	22	19.90	34.32	26	19.78	31.79
Rooster/0.90	33	43.17	28.71	40	50.77	28.75
Faculty/0.30	17	7.40	34.76	22	7.54	34.66
Faculty/0.50	20	13.61	31.88	24	13.41	31.80
Faculty/0.70	22	17.82	28.45	32	28.34	28.42
Faculty/0.90	35	36.10	24.20	47	45.01	24.10

Table 4.2: Performance comparison between OKB and HZ for different images and compression levels

4.3.1 Commentaries

From the results obtained in the tables above, and the performance profile results, as depicted in Figures (4.1), (4.2), (4.3) and (4.4) it is clear that our new algorithm based on the β_k^{OKB} parameter is more efficient than the HZ method in terms of number of iterations and computation time. There is a significant reduction in the number of iterations using the algorithm OKB compared to algorithm HZ. Furthermore, our algorithm is more competitive in terms of computation time with the HZ method. On the other hand, when the size of some examples becomes large, the HZ algorithms fail to provide the optimal solution after a number of iterations $k_{\max} = 50000$. As for image restoration, the OKB method gives an optimal balance between computing time and image quality, particularly in scenarios with high levels of noise. It achieves superior PSNR values and requiring fewer iterations than HZ approach.

4.4 Conclusion

We have proposed a new β_k, also we have provided proof of the global convergence of our proposed algrithm OKB for nonlinear functions using the strong Wolfe line search. The numerical test carried out confirm the effectiveness of the new algorithm OKB. It has good performance compared to Hager-Zhang (HZ) method for the selected test functions

and for the image restoration problems.

Conclusion

In this thesis, we presented a theoretical and numerical study of new conjugate gradient methods for solving unconstrained nonlinear optimization problems. Our main goal was to improve the performance and reliability of some conjugate gradient algorithms, especially when dealing with large scale or high dimensional problems.

We started by looking at well-known conjugate gradient methods like Fletcher-Reeves (FR) [15] and Hager-Zhang (HZ) [18]. While these methods are efficient in many cases, they often struggle with slow convergence or failure in high dimensions. To improve on these, we proposed two new conjugate gradient update formulas.

The first method, called MSD algorithm [30], is based on the FR formula given by:

$$\beta_k^{\text{MSD}} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2 + \mu|g_k^T d_{k-1}|}, \quad \mu > 0.$$

This helps the method stay stable and ensures that each step moves in a good direction.

The second method, called OKB algorithm [31], is inspired from the HZ approach. It uses both gradient information and past directions to improve convergence:

$$\beta_k^{\text{OKB}} = \frac{\|g_k\|^2 - \frac{\|g_k\|}{\|d_{k-1}\|}|g_k^T d_{k-1}|}{d_{k-1}^T y_{k-1}}, \quad \text{where } y_{k-1} = g_k - g_{k-1}.$$

This formula helps to avoid the common problem of the algorithm stalling or taking inefficient steps.

We proved that the two proposed directions satisfy the sufficient descent condition and both methods converge globally under strong Wolfe line search conditions.

We tested MSD and OKB algorithms on more than 100 standard test problems, with large sizes and we applied these approaches on image restoration problems. The obtained results were very promising:

- **MSD** reduced the number of iterations by 30–50% compared to FR and successfully solved problems that FR failed on.
- **OKB** was faster than HZ and more reliable in high dimensional cases.
- As for image restoration problems, our two new algorithms such that MSD and OKB offer a high quality images in compared to FR and HZ algorithms.

Both methods showed improvements in speed of convergence, stability, and robustness. The parameter μ in MSD algorithm made it more stable on tough problems, and both methods kept performing well as problem size increased.

Finally, this work shows that with carefully designed formulas and theoretical support, we can significantly improve classical conjugate gradient methods and make them more useful for modern large scale optimization problems.

Future Work

There are several interesting directions for future research:

1. **Stochastic optimization**: Adapting MSD and OKB to work with noisy or incomplete data, as found in machine learning problems.
2. **Combining with other methods**: Merging conjugate gradient steps with quasi-Newton methods BFGS to improve performance on more difficult problems.
3. **Real world applications**: Testing these methods on practical problems, such as neural network training, hyper-parameter tuning, or energy system optimization.
4. **Theoretical refinements**: Studying the worst case performance of our methods to better understand their behavior and limitations.

Bibliography

[1] N. Andrei, *An unconstrained optimization test functions collection*, Adv. Model. Optim. 10(1), 147–161 (2008).

[2] N. Andrei, *Nonlinear conjugate gradient methods for unconstrained optimization*, Springer Optimization and its Applications, vol. 158 (2020). DOI: https://doi.org/10.1007/978-3-030-42950-8.

[3] S. Ben Hanachi, B. Sellami and M. Belloufi, *New iterative conjugate gradient method for nonlinear unconstrained optimization*, RAIRO-Oper. Res. 56, 2315–2327 (2022). DOI: https://doi.org/10.1051/ro/2022109

[4] R. Benzine, *Optimisation sans contraintes, Tome 2*, Université de Annaba (2016).

[5] R. Chan, C. Ho and M. Nikolova, *Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization*, IEEE Transactions on Image Processing. 14, 1479–1485 (2005). DOI: https://doi.org/10.1109/TIP.2005.852196.

[6] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with performance profiles*. Math. Program. 91, 201–213 (2002). DOI: https://doi.org/10.1007/s101070100263

[7] Y. H. Dai and Y. Yuan, *A nonlinear conjugate gradient method with a strong global convergence property*, SIAM J. Optim. 10(1), 177–182 (1999). DOI: https://doi.org/10.1137/S1052623497318992

[8] Y. H. Dai and Y. Yuan, *An efficient hybrid conjugate gradient method for unconstrained optimization*, Ann. Oper. Res. 103, 33–47 (2001). DOI: https://doi.org/10.1023/A:1012930416777.

[9] Y. H. Dai and Z. Liao, *New conjugacy conditions and related nonlinear conjugate gradient methods*, Appl. Math. Optim. 43(1), 87–101 (2001). DOI: https://doi.org/10.1007/s002450010019.

[10] S. Delladji, M. Belloufi, B. Sellami, *New hybrid conjugate gradient method as a convex combination of FR and BA methods*, J. Inf. Optim. Sci. 42(3), 591–602 (2021).

[11] S. Djordjevic, *New hybrid conjugate gradient method as a convex combination of LS and FR methods*, Acta Math. Sci. 39B(1), 214–378 (2019). DOI: https://doi.org/10.2298/FIL1706813D.

[12] S. Djordjevic, *New hybrid conjugate gradient method as a convex combination of HS and FR methods*, J. Appl. Math. Comput. 2(9), 366–378 (2019). DOI: https://doi.org/10.1007/s10473-019-0117-6.

[13] H. Fan, Z. Zhu, A. Zhou, *A new descent nonlinear conjugate gradient method for unconstrained optimization*, Appl. Math. 2(9), 1119–1123 (2011). DOI: https://doi.org/10.4236/am.2011.29154.

[14] R. Fletcher, *Practical methods of optimization. Unconstrained Optimization*, 1, Wiley, New York (1987).

[15] R. Fletcher and C. M. Reeves, *Function minimization by conjugate gradients*, Comput. J. 7(2), 149–154 (1964). DOI: https://doi.org/10.1093/comjnl/7.2.149.

[16] N. Gould, D. Orban and P. Toint, *CUTEr and SifDec: A constrained and unconstrained testing environment, revisited*, ACM Transactions on Mathematical Software, 29(4), 373–394 (2003). DOI: https://doi.org/10.1145/962437.962438.

[17] A. Hallal, M. Belloufi, and B. Sellami, *An efficient new hybrid CG-method as convex combination of DY and CD and HS algorithms*, RAIRO-Oper. Res. 56, 4047–4056 (2022). DOI: https://doi.org/10.3934/mfc.2023028.

[18] W. W. Hager and H. Zhang, *A survey of nonlinear conjugate gradient methods*, Pac. J. Optim. 2, 35–58 (2006).

[19] M. R. Hestenes and E. Steifel, *Methods of conjugate gradients for solving linear systems*, J. Res. Natl. Bur. Stand. 49(6), 409–436 (1952). DOI: https://doi.org/10.6028/jres.049.044.

[20] Y. E. Hemici, S. Khelladi, and D. Benterki, *New hybrid conjugate gradient method for nonlinear optimization with application to image restoration problems*, Kybernetika, 60 (4), 535–552 (2024). DOI: http://doi.org/10.14736/kyb-2024-4-0535.

[21] B. Ivanov et al., *A novel value for the parameter in the Dai-Liao type conjugate gradient method*, J. Funct. Spaces, Article ID 6693401 (2021). DOI: http://doi.org/10.1155/2021/6693401.

[22] B. Ivanov et al., *New Hybrid Conjugate Gradient and Broyden–Fletcher–Goldfarb–Shanno Conjugate Gradient Methods*, J. Optim. Theory Appl. 178, 860–884 (2018). DOI: http://doi.org/10.1007/s10957-018-1324-3.

[23] S. Khelladi, *Optimisation sans contraintes, cours et exercices*, Université Ferhat Abbas Sétif 1 (2020).

[24] S. Khelladi and D. Benterki, *Hybrid conjugate gradient-BFGS methods based on Wolfe line search*, Stud. Univ. Babeș-Bolyai Math., 67(4), 855–869 (2022). DOI: http://dx.doi.org/10.24193/subbmath.2022.4.14

[25] S. Khelladi and D. Benterki, *Efficient one-parameter family of conjugate gradient methods*, J. Inf. Optim. Sci., 45(3), 699–715 (2024). DOI: https://doi.org/10.47974/JIOS-1354

[26] Y. Liu and C. Storey, *Efficient generalized conjugate gradient algorithms, part 1: theory*, J. Optim. Theory Appl. 69(1), 129–137 (1991). DOI: https://doi.org/10.1007/BF00940051.

[27] P. Mtagulwa and P. Kaelo, *A convergent modified HS-DY hybrid conjugate gradient method for unconstrained optimization problems*, J. Inf. Optim. Sci. 40(1), 97–113 (2019). DOI: https://doi.org/10.1080/02522667.2018.1424087.

[28] H. Y. Najm, E. T. Hamed and H. I. Ahmed, *Global convergence of conjugate gradient method in unconstrained optimization problems*, Bol. Soc. Paran. Mat. 38, 227–231 (2020). DOI: https://doi.org/10.5269/bspm.v38i7.46490.

[29] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer Series in Operations Research and Financial Engineering (2006).

[30] M. Ouaoua and S.Khelladi, *Efficient Descent Direction of a Conjugate Gradient Algorithm for Nonlinear Optimization*, Nonlinear Dyn. Syst. Theory. 25(1), 91–100 (2025).

[31] M. Ouaoua, S.Khelladi and D. Benterki, *New parameter of conjugate gradient method for unconstrained nonlinear optimization*, Stat., Optim. Inf. Comput., 13(6), (2025). DOI: https://doi.org/10.19139/soic-2310-5070-2069.

[32] B. T. Polyak, *The conjugate gradient method in extrem problems*, U.S.S.R. Comput. Math. Phys. 9, 94–112 (1969).

[33] E. Polak and G. Ribiere, *Note sur la convergence des méthodes de directions conjuguées*, Rev. Française Inf. Rech. Opér. 16, 35–43 (1969).

[34] N. Rahali, *Accélération de la convergence de quelques méthodes d'optimisation sans contraintes*, Université Ferhat Abbas Sétif 1 (2021).

[35] M. Rivaie, M. Mustafa, W. J. Leong, M. Ismail, *A new class of nonlinear conjugate gradient coefficients with global convergence properties*, Appl. Math. Comput. 218(22), 11323–11332 (2012). DOI: https://doi.org/10.1016/j.amc.2015.07.019.

[36] B. Sellami and Y. Chaib, *A new family of globally convergent conjugate gradient methods*, Ann. Oper. Res. 241, 497–513 (2016). DOI: https://doi.org/10.1007/s10479-016-2120-9.

[37] B. Sellami and Y. Chaib, *New conjugate gradient methods for unconstrained optimization*, Rairo Operations Research. 50, 1013–1026 (2016). DOI: https://doi.org/10.1051/ro/2015064.

[38] W. Sun and Y. Yuan, *Optimization theory and methods Nonlinear programming*, SOIA. 1, 120–150 (2006). DOI: https://doi.org/10.1007/b106451.

[39] Z. Wei, S. Yao, L. Liu, *The convergence properties of some new conjugate gradient methods*, Appl. Math. Comput. 183, 1341–1350 (2006). DOI: https://doi.org/10.1016/j.amc.2006.05.150.

[40] Q. J. Wu, *A nonlinear conjugate gradient method without line search and its global convergence*, Int. Conf. Comput. Inf. Sci., 1148–1152 (2011). DOI: `https://doi.org/10.1109/ICCIS.2011.45`.

[41] S. Yao, X. Lu, and Z. Wei, *A conjugate gradient method with global convergence for large-scale unconstrained optimization problems*, J. Appl. Math., Article ID 730454 (2013). DOI: `https://doi.org/10.1155/2013/730454`.

[42] G. Zoutendijk, *Nonlinear programming, computational methods*, In: J. Abadie (ed.) Integer and Nonlinear Programming, 37–86 (1970).

ملخص :
تتناول هذه الأطروحة حل مسائل الأمثلة غير الخطية وغير المقيدة باستخدام طرق التدرج المترافق، حيث اقترحنا طريقتين جديدتين من نوع التدرج المترافق، و قمنا بإثبات شرط التناقص بالنسبة لاتجاهات البحث الجديدة الناتجة وبرهنا التقارب الإجمالي للخوارزميات المقترحة باستخدام البحث الخطي لوولف.
كما تم تعزيز النتائج المحصل عليها من خلال تجارب عددية مختلفة ذات أبعاد كبيرة، أثبتت تفوق الطرق المقترحة مقارنة مع طرق أخرى.
كلمات مفتاحية :
الأمثلة غير الخطية غير المقيدة، طريقة التدرج المترافق، اتجاه التناقص، البحث الخطي غير الدقيق، التقارب الإجمالي.

Abstract:
This thesis deals with solving unconstrained nonlinear optimization problems using conjugate gradient methods. We proposed two new algorithms of the conjugate gradient type, based on two new conjugacy parameters. We established the descent condition for the new search directions, and we proved the global convergence of the corresponding algorithms, under the Wolfe line search.
The effectiveness of the proposed algorithms is confirmed through numerical tests carried out on several large-scale numerical experiments.
Keywords:
Unconstrained nonlinear optimization, Conjugate gradient method, Descent direction, Inexact line search, Global convergence.

Résumé:
Cette thèse concerne la résolution des problèmes d'optimisation non linéaire sans contraintes, en utilisant les méthodes de gradient conjugué. Nous avons proposé deux nouvelles direction de recherche de méthode de gradient conjugué issues de deux nouveaux paramètres de conjugaison. Nous avons démontré la condition de descente de ces directions de recherche, ainsi que la convergence globale des algorithmes correspondants en utilisant la recherche linéaire de Wolfe.
L'efficacité des algorithmes proposés est confirmée par des tests numériques réalisés sur plusieurs problèmes de grandes dimensions.
Mots clés :
Optimisation non linéaire sans contraintes, Méthode du gradient conjugué, Direction de descente, Recherche linéaire inexacte, Convergence globale.