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

Setif 1 University Ferhat Abbas
Faculty of Sciences
Department of Mathematics

جامعة سطيف 1 فرحات عباس
كلية العلوم
قسم الرياضيات

Doctoral Thesis

Submitted in Fulfillment of the Requirements for the Degree of Doctor in Mathematics

Option : Nonlinear Optimization

Theme :

APPLICATION OF INTERIOR POINT METHODS FOR NON-LINEAR PROGRAMMING

Presented by

Miss. **SOULI Choubeila**

Supervisor : **Assma LEULMI**

In front of the jury composed of :

Mohamed ACHACHE	Pr	Univ. Ferhat Abbas Setif1	President
Assma LEULMI	MCA	Univ. Ferhat Abbas Setif1	Supervisor
Raouf ZIADI	MCA	Univ. Ferhat Abbas Setif1	Co-supervisor
Nasreddine HAMRI	Pr	Univ. Abdelhafid Boussouf Mila	Examiner
Tayeb HAMAIZIA	Pr	Univ. Frères Mentouri Constantine1	Examiner
Louiza DERBAL	MCA	Univ. Ferhat Abbas Setif1	Examiner

2025


بسم الله الرحمن الرحيم

Life is good only for two things, discovering mathematics and teaching mathematics

SIMEON DENIS POISSON

Dedicate

This work is dedicated to

my beloved parents, whose endless love, sacrifices, and encouragement have been the foundation of my success.

To my dear family and friends, who have always been by my side, providing unwavering support and joy throughout this journey.

And finally, to myself for the perseverance, dedication, and strength to overcome challenges and achieve this milestone.

CHOUBEILA SOULI.

Acknowledgment

First and foremost, we would like to express our deep gratitude to Allah Almighty, who granted us the will, patience, strength, and health necessary to complete this modest work.

I would like to express my special gratitude to my supervisor, Asmaa LEULMI, and the co-supervisor, Raouf ZIADI, for their guidance throughout the years of preparing my thesis. Their constant dedication to the progress of my research has been invaluable. The numerous discussions we had, along with their wise advice, have significantly contributed to the final outcome of this work.

I would also like to express my appreciation and respect to Mr. Mohamed ACHACHE, Professor at Setif 1 University, who honored me by presiding over the thesis defense committee. My sincere thanks also go to the committee members: Mr. HAMRI Nasreddine, from Abdelhafid Boussouf University Center of Mila; Mr. HAMAIZIA Tayeb, from Frères Mentouri Constantine 1 University; and Mrs. DERBAL Louiza,from Setif 1 University. I greatly appreciate their interest in my research and their willingness to examine my work and enrich it with their valuable comments.

CHOUBEILA SOULI.

List of Tables

List of Figures

List of Algorithms

Notation and Symbols

The symbols and notations used in this thesis are outlined below

Notations

- **MP**: Mathematical programming.
- **NLP**: Nonlinear programming.
- **IPMs**: Interior point methods.
- **CGMs**: Conjugate gradient methods

Symbols

- $\mathbb{R}$: The set of real numbers.
- $\mathbb{R}^+$: The set of positive real numbers.
- $\mathbb{R}^n$: The euclidean vector space of dimension n.
- $\mathbb{R}^n_+$: The positive orthant in $\mathbb{R}^n$.
- (a,b): The open interval of real numbers from a to b.
- $[a,b]$: The closed interval of real numbers from a to b.

- $Dom(h)$: The effective domain of $h : S \to \mathbb{R}$ can be expressed as

$$\text{Dom}(h) = \{x \in S : h(x) < \infty\}.$$

- $x \geq 0$: Implies that every component x_i of the vector x is greater than or equal to zero.
- x_k: The k^{th} vector in a sequence of vectors.
- x_i: The i^{th} component of x.
- $x^\top$: The transpose of the vector x.
- $\{e_1, \ldots, e_n\}$: The canonical basis in $\mathbb{R}^n$.
- $A \in \mathbb{R}^{m \times n}$: A matrix consisting of m rows and n columns.
- $\text{Int}(S)$: The interior of a set S.
- m: The number of constraints in the nonlinear optimization problem, which is also the number of rows in the constraint matrix A.
- n: The number of decision variables in the nonlinear optimization problem, corresponding to the number of columns in the constraint matrix A.
- $\|x\|$: The Euclidean norm of the vector x.
- $\langle \cdot, \cdot \rangle$: The usual scalar product in $\mathbb{R}^n$.

Contents

Introduction

In the modern era, optimization problems have become a cornerstone for addressing numerous scientific, engineering, and economic challenges. These problems focus on finding the optimal values for objective functions that represent the performance of systems or processes, while considering constraints that define the nature of the problem. Nonlinear optimization problems, in particular, are among the most complex types, characterized by nonlinear objective functions or constraints, making their resolution a challenge that requires advanced tools and techniques.

In recent decades, significant progress has been made in developing innovative methods to solve nonlinear optimization problems, thanks to advancements in applied mathematics and computing. Among the most prominent methods, conjugate gradient methods [53, 54, 46, 47] and interior point methods [2, 4, 5, 6, 31, 32].

Conjugate gradient methods (CGMs) have emerged as indispensable tools for solving nonlinear unconstrained optimization problems, particularly those involving large-scale systems. These methods strike a balance between computational efficiency and low memory requirements, making them especially suitable for problems where direct methods, such as Newton's method, become impractical. CGMs are particularly appealing due to their iterative nature, leveraging conjugacy principles to achieve faster convergence compared to traditional gradient descent methods. The origins of CGMs date back to 1952, when Hestenes and Stiefel [34] introduced the CG method to solve symmetric positive definite linear equation systems. In the 1960s, Fletcher and Reeves [27] refined this method, resulting in the Fletcher-Reeves (FR) method and extending it to address unconstrained nonlinear optimization problems. Various conjugate gradient methods are associated with specific choices for the scalar parameter

β_k. In 1969, Polak and Ribière [53] and Polyak [54] independently proposed a conjugate gradient method that later became known as the Polak-Ribière-Polyak (PRP) method. Fletcher introduced a variant called the Conjugate Descent (CD) method [26], while Liu and Storey proposed the Liu-Storey (LS) method [45]. Additionally, Dai and Yuan [19] examined and refined the approach, culminating in the Dai-Yuan (DY) method. The most important characteristics of CG methods are their global convergence and numerical performance. According to [19, 26, 27, 34, 45, 53, 54], the aforementioned methods can be broadly categorized into two classes. The FR, CD, and DY methods exhibit excellent global convergence properties but demonstrate relatively weaker practical performance. Conversely, the HS, PRP, and LS methods exhibit superior numerical performance but may not always guarantee convergence. To address these limitations, new hybrid conjugate gradient methods have been proposed in the literature [53, 56, 50, 62, 47, 46]. These hybrid methods aim to combine robust global convergence properties with improved practical performance, offering a more balanced and effective approach for solving nonlinear optimization problems.

On the other hand, one of the most advanced and effective approaches for solving nonlinear programming (NLP) problems is the family of Interior Point Methods (IPMs). These methods have gained widespread recognition for their ability to handle large-scale, highly constrained optimization problems. By navigating through the interior of the feasible region, IPMs efficiently converge toward optimal solutions without being overly sensitive to the problem's size or condition. Despite their established success, challenges remain in further improving computational efficiency, particularly for high-dimensional nonlinear problems. Interior Point Methods were first introduced in 1955 by K. R. Frisch [28] to solve convex programming problems, demonstrating polynomial complexity. In 1967, P. Huard [35] proposed the method of centers for solving problems with nonlinear constraints. Subsequently, in 1968, A. V. Fiacco and G. P. McCormick [25] extended IPMs to address convex nonlinear programming problems. In 1970, N. Shor [58] introduced the ellipsoid method for linear programming, which was further developed in 1979 by L. G. Kachian [39], who established its polynomial complexity. Many studies have focused on using IPMs to solve linear, quadratic, semidefinite, and nonlinear programming problems, with significant attention given to logarithmic barrier methods. These methods, initially proposed by Frisch and later developed by

Fiacco and McCormick, reformulate the non-negativity constraints $x_i \geq 0$ into a penalty term. As the barrier parameter approaches zero, the convergence of the solution can be mathematically proven. This concept of transforming constrained problems into unconstrained ones is rooted in the broader framework of penalization. Penalization serves as a powerful approach to simplify optimization problems by introducing penalty terms that account for constraints. When the penalty function is carefully designed, the penalized problem can retain the properties of the original problem, a concept known as exact penalization. However, when such direct correspondence is not achievable, inexact penalization employs limiting processes to recover the desired properties, ensuring that the original problem's characteristics are ultimately reflected.

This thesis aims to study and analyse nonlinear optimization problems and review various methods for solving them, with a particular focus on conjugate gradient methods and interior point methods based on barrier function. The theoretical concepts of these methods will be highlighted, along with their practical applications and performance evaluation in solving a variety of problems. This work underscores the importance of nonlinear optimization as a pivotal tool for improving performance and developing innovative solutions across various scientific and industrial fields.

This work is composed of four chapters.

- In the first chapter, we introduce fundamental concepts in convex analysis, optimization, and mathematical programming. We discuss the existence and uniqueness of optimal solutions and explore various resolution methods for mathematical programming, including conjugate gradient methods and interior point methods. These foundational concepts and techniques serve as a basis for the subsequent chapters, where they are applied to establish the theoretical results.

- In the second chapter, we focus on solving a nonlinear optimization problem (NLP) using a barrier logarithmic penalty method. To perturb the NLP, we propose using a logarithmic barrier function of the form $\sum_{i=1}^{n} \rho_i \ln(\rho_i) - \sum_{i=1}^{n} \rho_i \ln(x_i)$, where $\rho \in \mathbb{R}^n_+$. We then demonstrate the existence and uniqueness of the optimal solution for the perturbed problem, as well as analyse the convergence of the perturbed solution to the original problem. We compute the Newton descent direction and determine the step-

length using a minorant function technique, which is related to a secant method. Finally, we validate the effectiveness of our method through highly encouraging numerical simulations.

This study has been accepted for publication in **" the Journal of Nonlinear Dynamics and Systems Theory"**

- In the third and fourth chapters, we introduce two hybrid conjugate gradient methods. For each method, we outline the procedure, analyse the sufficient descent property, prove global convergence, and assess their efficiency and reliability based on numerical performance. Furthermore, we explore the application of these methods in image restoration problems.

  These studies resulted in articles [60, 61] that were published in **"the Journal of Mathematical Modeling" and "the Iranian Journal of Numerical Analysis and Optimization"**.

CHAPTER 1

Preliminaries and Fundamental Concepts

In this chapter, we will outline several properties and definitions of essential concepts that are crucial for demonstrating the theoretical results in the subsequent chapters (see [9, 12, 14, 17, 38, 51, 55]).

1.1 Convex Analysis

1.1.1 Affine sets and functions

Definition 1.1.1 ***(Affine set).***

A subset S of $\mathbb{R}^n$ is considered affine if

$$\forall x, z \in S, \lambda x + (1-\lambda) z \in S, \forall \lambda \in \mathbb{R}.$$

Examples of basic affine sets are the empty set $(\emptyset)$, singletons $(\{x\})$ where $x \in \mathbb{R}^n$, and all vector sub-spaces of $\mathbb{R}^n$.

Definition 1.1.2 ***(Affine function).***

We define a function $h : \mathbb{R}^n \to \mathbb{R}$ as affine if, for any vectors x and z in $\mathbb{R}^n$ and any scalar $\lambda \in \mathbb{R}$, the following condition holds

$$h(\lambda x + (1-\lambda) z) = \lambda h(x) + (1-\lambda) h(z).$$

1.1.2 Convexity concepts

Definition 1.1.3 ***(Convex set)***.

A subset S of $\mathbb{R}^n$ is considered convex if

$$\forall x, z \in S;\ \ \lambda x + (1-\lambda) z \in S,\ \ \forall \lambda \in [0,1].$$

This definition means that a set S is convex if it contains the line segment connecting any two points x and z within S, i.e.

$$[x,z] = \{\lambda x + (1-\lambda) z : \lambda \in [0,1]\} \subseteq S.$$

Definition 1.1.4 ***(Convex cones)***.

A subset C of $\mathbb{R}^n$ is a cone if and only if, for any $x \in C$ and any $\lambda > 0$, we have $\lambda x \in C$.

. *C is called a pointed (or salient) cone if $C \cap (-C) = \{0\}$.*

. *C is called a convex cone if the set C is convex.*

Definition 1.1.5 ***(Cone of recession)***.

The recession cone of C is the set

$$C_\infty = \bigcap_{a \in C} C_\infty(a_1).$$

where $C_\infty(a) = \{d \in \mathbb{R}^n : a + \lambda d \in C, \forall \lambda > 0\}$ is a non-empty convex cone and $d \in C_\infty$ is called a direction of recession.

Theorem 1.1.1 *If $C \neq \emptyset$ is convex and closed, then $C_\infty(a_1) = C_\infty(b_1)$ for all $a_1, b_1 \in C$.*

Theorem 1.1.2 *A function $h : \mathbb{R}^m \to \mathbb{R} \cup \{\infty\}$ is said to be inf-compact if and only if $C_\infty(h) = \{0\}$.*

Proposition 1.1.0 *Let C be a non-empty closed convex set in $\mathbb{R}^n$, then*

$$(C \text{ is bounded}) \iff (C_\infty = \{0\}).$$

Definition 1.1.6 ***(Convex function)***.

A real function h, defined on a convex set $S \subseteq \mathbb{R}^n$, is convex if

$$\forall x, z \in S, \forall \lambda \in [0,1], h(\lambda x + (1-\lambda)z) \leq \lambda h(x) + (1-\lambda)h(z).$$

Definition 1.1.7 ***(Strictly convex function)***.

A real function h, defined on a convex set $S \subseteq \mathbb{R}^n$, is strictly convex if

$$\forall x, z \in S, x \neq z, \forall \lambda \in (0,1), h(\lambda x + (1-\lambda)z) < \lambda h(x) + (1-\lambda)h(z).$$

Definition 1.1.8 ***(Strongly convex function)***.

A real function h is said to be strongly convex with a coefficient $\mu > 0$ if the following condition holds for all $x, z \in S$ and $\lambda \in [0,1]$

$$h(\lambda x + (1-\lambda)z) \leq \lambda h(x) + (1-\lambda)h(z) - \frac{\mu}{2}\lambda \|x-z\|^2.$$

Definition 1.1.9 ***(Proper function)***.

A real function h, defined on a set $S \subseteq \mathbb{R}^n$, is called a proper function if it satisfies the following conditions

- $h(x) > -\infty$, $\forall x \in S$.
- $dom(h) \neq \emptyset$.

where $dom(h)$ denotes the effective domain of h.

Definition 1.1.10 ***(Convex combination)***.

A vector $z \in S \subseteq \mathbb{R}^n$ is a convex combination of the points $\{x_1, x_2, ..., x_p\}$ if there exist real coefficients $\lambda_i \geq 0$, $i \in \{1, ..., p\}$, such that

$$z = \sum_{i=1}^{p} \lambda_i x_i \text{ with } \sum_{i=1}^{p} \lambda_i = 1.$$

Definition 1.1.11 ***(Convex hull)***.

The convex hull of a set $S \subseteq \mathbb{R}^n$ is the set of points in $\mathbb{R}^n$ that can be written as convex combinations of points from S. i.e.,

$$conv(S) = \{x \in \mathbb{R}^n : x = \sum_{i=1}^{p} \lambda_i x_i, x_i \in S, \lambda_i \geq 0, \forall i \in \{1, ..., p\} \text{ and } \sum_{i=1}^{p} \lambda_i = 1\}.$$

Definition 1.1.12 ***(Convex polyhedron)***.

A subset S is defined as a convex polyhedron if it is the intersection of finitely many half-spaces in $\mathbb{R}^n$. It can be expressed as

$$S = \bigcap_{i=1}^{n} \{x \in \mathbb{R}^n : \langle b_i, x \rangle \leq c, b_i \in \mathbb{R}^n, c \in \mathbb{R}\}.$$

Definition 1.1.13 ***(Extreme point)***.

Let S be a non-empty convex set in $\mathbb{R}^n$. A point x is defined as an extreme point (or vertex) of S if

$$\forall x_1, x_2 \in S, \lambda \in]0,1[, (x = \lambda x_1 + (1-\lambda) x_2) \implies (x = x_1 = x_2).$$

Every extreme point of a convex set S lies on its boundary.

Proposition 1.1.0 *Let S be a non-empty convex set in $\mathbb{R}^n$. A point x is a vertex of S if and only if $S \setminus \{x\}$ is a convex set.*

1.1.3 Differentiability and convexity

Definition 1.1.14 *The gradient of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$, at a point $x \in \mathbb{R}^n$, is defined by*

$$\nabla h(x) = \left(\frac{\partial h(x)}{\partial x_1}, \frac{\partial h(x)}{\partial x_2}, ..., \frac{\partial h(x)}{\partial x_n} \right)^\top.$$

The elements of the Hessian matrix at the i^{th} row and j^{th} column are defined as

$$\nabla^2 h(x)_{i,j} = \left(\frac{\partial^2 h(x)}{\partial x_i \partial x_j} \right)_{i,j}, \quad i, j = \{1, .., n\}.$$

Definition 1.1.15 *If a function $h \in \mathscr{C}^1$ and S is a convex set, the following equivalences hold*

1. *h is convex if and only if, $h(x) - h(z) \geq \langle \nabla h(z), x - z \rangle, \quad \forall x, z \in S.$*

2. *h is convex if and only if, $\langle \nabla h(x) - \nabla h(z), x - z \rangle \geq 0, \quad \forall x, z \in S.$*

Moreover, h is strictly convex if either of the above inequalities holds strictly whenever $x \neq z$.

Proposition 1.1.0 *Let $h : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function defined on a convex domain D.*

1. *h is a convex function on D if and only if the Hessian matrix $\nabla^2 h(x)$ is positive semidefinite, i.e.,*

$$\forall x \in D : z^\top \nabla^2 h(x) z \geq 0, \forall z \in \mathbb{R}^n,$$

or equivalently, all eigenvalues of $\nabla^2 h(x)$ are non-negative.

2. *h is a strictly convex function on D if and only if the Hessian matrix $\nabla^2 h(x)$ is positive definite, i.e.,*

$$\forall x \in D : z^\top \nabla^2 h(x) z > 0, \forall z \in \mathbb{R}^n, z \neq 0,$$

or equivalently, all eigenvalues of $\nabla^2 h(x)$ are strictly positive.

Lemma 1.1.1 *A function h is called concave if $-h$ is convex.*

Definition 1.1.16 *A function h is called coercive on a convex set S if*

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

Definition 1.1.17 ***(Admissible direction).***

Let S be a non-empty subset of $\mathbb{R}^n$. A vector $d \neq 0$ in $\mathbb{R}^n$ is called a direction of S at $c \in S$ if

$$\text{There exists } \beta_0 > 0 \text{ such that } c + \beta d \in S \text{ for all } \beta \in [0, \beta_0].$$

If this property holds for all $c \in S$, d is referred to as an admissible direction for S.

Definition 1.1.18 ***(Directional derivative).***

Let h be a function from $\mathbb{R}^n$ to $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$, and let x_0 be a point in $\mathbb{R}^n$ such that $h(x_0)$ takes a finite value. The directional (one-sided) derivative of h at x_0 in the direction $d \in \mathbb{R}^n$ is defined by

$$h'_d(x_0) = \lim_{\tau \to 0^+} \frac{h(x_0 + \tau d) - h(x_0)}{\tau}$$

if it exists. (The limit may take values $\pm\infty$).

Note that

- $-h'_{-d}(x_0) = \lim_{\tau \to 0^-} \frac{h(x_0 + \tau d) - h(x_0)}{\tau}.$

- *The directional derivative is called bilateral if* $h'_d(x_0) = -h'_{-d}(x_0)$.
- *If* $h'_d(x_0)$ *exists for all* $d \in \mathbb{R}^n$, *we state that* h *is directionally differentiable at* x_0.
- *If there exists a constant vector* $y_0 \in \mathbb{R}^n$ *such that* $h'_d(x_0) = \langle y_0, d\rangle$ *for all* $d \in \mathbb{R}^n$, *we define* h *is G-differentiable (or differentiable in the Gateaux sense) at* x_0, *denoted by* $h'(x_0) = y_0$.

1.1.4 Lower and upper semicontinuous functions

Definition 1.1.19 ***(Lower semicontinuous function)***.

A function $h : \mathbb{R}^n \to \mathbb{R}$ *is called lower semicontinuous at* $x_0 \in \mathbb{R}^n$ *if*

$$\text{for every } \epsilon > 0, \text{ there exists } \delta > 0 \text{ such that } h(x) \geq h(x_0) - \epsilon \text{ whenever } \|x - x_0\| \leq \delta.$$

Definition 1.1.20 ***(Upper semicontinuous function)***.

A function h *is called upper semicontinuous at* x_0 *if* $-h$ *is lower semicontinuous at* x_0.

Remark 1.1.1 *A function* h *is continuous at* x_0 *if and only if it is both lower and upper semicontinuous at* x_0.

1.2 Mathematical Programming

An optimization problem consists of a given set $S \subseteq \mathbb{R}^n$ (the set of actions or strategies) and a function h (the criterion or objective function) defined on this set. The goal is to determine whether there exists an action in S that optimizes h.

From a mathematical perspective, the problem is formulated as

- unconstrained optimization problems: $\min_{x \in \mathbb{R}^n} h(x) \quad (P_1)$
- constrained optimization problems: $\min_{x \in S} h(x) \quad (P_2)$

1.2.1 Unconstrained optimization problems

A general unconstrained mathematical programming problem can be expressed as follows

$$\min_{x\in\mathbb{R}^n} h(x). \tag{P_1}$$

The problem (P_1) has a solution if there exists a variable $x^* \in \mathbb{R}^n$ such that

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

We say that x^* is a minimizer (or a minimum point) of h on $\mathbb{R}^n$ and that $h(x^*)$ is a minimum of h on $\mathbb{R}^n$.

Definition 1.2.1 *Let $h : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function.*

- *A point $x^* \in \mathbb{R}^n$ is a global minimum of problem (P_1) if and only if*

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

- *A point $x^* \in \mathbb{R}^n$ is a local minimum of (P_1) if and only if there exists an open neighborhood $V_\epsilon(x^*)$ of x^* such that*

$$\forall x \in V_\epsilon(x^*),\ h(x^*) \leq h(x).$$

- *A point $x^* \in \mathbb{R}^n$ is a strict local minimum of (P_1) if and only if there exists an open neighborhood $V_\epsilon(x^*)$ of x^* such that*

$$\forall x \in V_\epsilon(x^*) \text{ where } x \neq x^*,\ h(x^*) < h(x).$$

Remark 1.2.1 *Any global minimum is also a local minimum, but the converse is not necessarily true.*

Existence and uniqueness results

Theorem 1.2.1 ***(Weierstrass)***.

If a subset $S \subset \mathbb{R}^n$ is compact, and h is continuous on S, then h admits at least one minimum on S. i.e.

$$\exists x^* \in S : h(x^*) = \min_{x \in S} h(x) \iff \exists x^* \in S, h(x^*) \leq h(x), \forall x \in S.$$

Theorem 1.2.2 ***(Existence)***.

If $h : \mathbb{R}^n \to \mathbb{R}$ is continuous and coercive function on $\mathbb{R}^n$, then h admits at least one minimum x^* on $\mathbb{R}^n$.i.e.

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

Theorem 1.2.3 ***(Sufficient condition for uniqueness)***.

If $h : \mathbb{R}^n \to \mathbb{R}$ is strictly convex function, then there exists at most one $x^* \in \mathbb{R}^n$ (global minimum).

Theorem 1.2.4 ***(Existence and uniqueness)***.

Let $h : \mathbb{R}^n \to \mathbb{R}$ be a function. Assume that

1. *h is continuous.*
2. *h is coercive.*
3. *h is strictly convex.*

Then (P_1) admits a unique global optimal solution x^*.

Conditions of optimality

1. **Necessary conditions (NC):**

   Theorem 1.2.5 ***(First-order NC)***.

   Let $h : \mathbb{R}^n \to \mathbb{R}$ be continuous and differentiable function at $x^* \in \mathbb{R}^n$. If x^* is either a global or local minimum of h, then $\nabla h(x^*) = 0$.

Theorem 1.2.6 ***(Second-order NC).***

Let $h : \mathbb{R}^n \to \mathbb{R}$ be continuous and twice differentiable function at $x^* \in \mathbb{R}^n$. If x^* is either a global or local minimum of h, then

- $\nabla h(x^*) = 0$,
- *The Hessian matrix $H(x^*)$ of h at x^* is positive semidefinite, i.e.,*

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

Remark 1.2.2

- *In the one-dimensional case (i.e., $h : \mathbb{R} \to \mathbb{R}$), we have*

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

- *A point $x^* \in \mathbb{R}^n$ which verifies $\nabla h(x^*) = 0$ is a stationary point or critical point of $h(x)$ in $\mathbb{R}^n$ (a candidate point to be a minimum).*
- *If h is differentiable and $\nabla h(x^*) \neq 0$, then x^* cannot be a minimum of h*

2. **Sufficient conditions (SC):**

Theorem 1.2.7 *Let $h : \mathbb{R}^n \to \mathbb{R}$ be a function twice differentiable at $x^* \in \mathbb{R}^n$, if*

1. $\nabla h(x^*) = 0$,
2. *$H(x^*)$ positive definite $(H(x^*) = \nabla^2 h(x^*))$.*

then, x^* is a local minimum of h,

Remark 1.2.3 *In the case of a single variable, we have*

$$\left.\begin{array}{l} h'(x^*) = 0 \\ h''(x^*) > 0 \end{array}\right\} \Rightarrow x^* \text{ is a global minimum of } h \text{ in } \mathbb{R}.$$

1.2.2 Constrained optimization problems

Let S be a non-empty subset of $\mathbb{R}^n$. Consider an optimization problem in the following form

$$\{\min h(x), x \in S\}. \tag{P_2}$$

Here, the function $h : \mathbb{R}^n \to \mathbb{R}$ is referred to as the cost function, the objective, or the criterion. The set $S = \{x \in \mathbb{R}^n : g_j(x) \leq 0, k_i(x) = 0, j = \overline{1,m}, i = \overline{1,p}\}$ is called the constraint set or the feasible set and $g_j, k_i : \mathbb{R}^n \to \mathbb{R}$ are given functions.

Any point $x \in \mathbb{R}^n$ satisfying $x \in S$ is called an admissible point or feasible point of problem (P_2).

Definition 1.2.2 *A feasible solution of (P_2) is any point in S that satisfies all the constraints.*

A global optimal solution x^* of (P_2) is a feasible solution that minimizes the objective function over S. Such a solution is denoted as

$$x^* = \arg\min_{x \in S} h(x).$$

A local optimal solution $x^* \in S$ of (P_2) is a feasible solution where there exists a neighborhood V of x^* such that

$$h(x^*) \leq h(x), \forall x \in S \cap V.$$

Existence and uniqueness results

Theorem 1.2.8 ***(Existence)***.

If $h : \mathbb{R}^n \to \mathbb{R}$ is a continuous and coercive function on S, then h has at least one minimum x^* on S i.e.,

$$\exists x^* \in \mathbb{R}^n : h(x^*) \leq h(x), \forall x \in S.$$

Theorem 1.2.9 *$h : \mathbb{R}^n \to \mathbb{R}$ has an optimal solution if and only if the recession cone C_d of h reduces to the origin i.e. $C_\infty(h) = \{0\}$*

Theorem 1.2.10 ***(Uniqueness)***.

If S is a non-empty convex set and h is a strictly convex function, then (P_2) has at most an optimal solution $x^* \in S$.

Qualification of constraints

Definition 1.2.3 ***(Feasible point).***

A point $x \in \mathbb{R}^n$ is called feasible for problem (P_2) if it satisfies all the constraints.

Definition 1.2.4 ***(Active constraints).***

Let $g : \mathbb{R}^n \to \mathbb{R}$. An inequality constraint $g(x) \leq 0$ is said to be active (or inactive) at x^* if $g(x^*) = 0$ (or $g(x^*) < 0$, respectively).

An equality constraint $k_i(x) = 0$ is always active for every feasible point $x \in S$.

Definition 1.2.5 *Constraints are considered qualified at any feasible point $x \in S$ if the following conditions hold*

- *S is convex and the interior of S is non-empty (Slater's condition (1950)).*
- *S is a convex polyhedron (with k_i and g_j being affine functions) (Karlin (1959)).*
- *If the gradients of all active constraints at $x^* \in S$ are linearly independent, then the constraints are qualified at x^* (Mangasarian-Fromovitz (1967)).*

Optimality conditions

The Lagrangian of problem (P_2) is defined as follows

$$L(x;\lambda;\mu) = h(x) + \sum_{j=1}^{m} \lambda_j k_j(x) + \sum_{i=1}^{p} \mu_i g_i(x),$$

where μ_i and λ_j are the Lagrange multipliers, where $\mu_i \in \mathbb{R}_+$ for all $i = 1,\ldots,p$, and $\lambda_j \in \mathbb{R}$ for all $j = 1,\ldots,m$. The Karush-Kuhn-Tucker (KKT) theory establishes necessary optimality conditions for constrained optimization problems with differentiable objective functions.

Theorem 1.2.11 *If x^* is a local optimal solution of (P_2) and satisfies one of the qualification conditions, then there exist multipliers, $\lambda \in \mathbb{R}^m$ and $\mu \in \mathbb{R}^p_+$ such that*

$$\begin{cases} \nabla h(x^*) + \sum_{j=1}^{m} \lambda_j \nabla k_j(x^*) + \sum_{i=1}^{p} \mu_i \nabla g_i(x^*) = 0 \quad \textit{(stationarity condition)} \\ \mu_i g_i(x^*) = 0, \quad i = \overline{1,p}, \quad \textit{(complementarity condition)} \\ k_j(x^*) = 0, \quad j = \overline{1,m} \end{cases}$$

Remark 1.2.4

- *If (P_2) is convex, the KKT conditions are necessary and sufficient for x^* to be a global minimum.*
- *If (P_2) is convex, every local minimum is a global minimum*
- *If the constraints are not qualified at x^*, the KKT conditions may not hold, which means x^* could still be optimal without satisfying them.*

1.3 Conjugate Gradient Methods

This section focuses on exact and approximate line search methods. It provides definitions, algorithms, and properties related to these types of line searches. Additionally, it defines the Newton method, the gradient method, and the two cases of the conjugate gradient method.

1.3.1 Descent direction methods

A general framework for a descent direction method is outlined as follows

- Begin with an initial point $x_0 \in \mathbb{R}^n$.
- For each $k \geq 0$: $x_{k+1} = x_k + \alpha_k d_k$,

where $\alpha_k \in \mathbb{R}_*^+$ represents the step-length. The descent direction d_k is chosen such that the condition

$$h(x_k + \alpha_k d_k) \leq h(x_k),$$

holds.

Theorem 1.3.1 *Consider a function h that is continuously differentiable in a neighborhood of $x_k \in \mathbb{R}^n$ and let $d_k \in \mathbb{R}^n$ be a non-zero vector. If the condition*

$$\nabla h(x_k)^\top d_k < 0,$$

is satisfied, then d_k is a descent direction for h at x_k.

Proof. For the proof, we refer to [49] ■

Algorithm 1: Descent direction algorithm

Step 0: Provide x_0, $\epsilon > 0$ and set $k = 0$.

Step 1: Compute d_k

Step 2: Solve $\min_{\alpha \in \mathbb{R}_+^*} h(x_k + \alpha d_k)$ for the step-length α_k, using either an exact or inexact line search.

Step 3: Set $x_{k+1} = x_k + \alpha_k d_k$.

Step 4: Check the stopping criterion: If $\|x_{k+1} - x_k\| \leq \epsilon$, stop the algorithm. Otherwise set $k = k + 1$ and go to **Step 1**.

Newton's method

The core concept of Newton's method for unconstrained optimization is the iterative application of a quadratic approximation, denoted by $q^{(k)}$ for the objective function h. This approximation is constructed at the current iteration point x_k, with the goal of minimizing $q^{(k)}$. Assume that the function $h : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable, with $x_k \in \mathbb{R}^n$, and that the Hessian matrix $\nabla^2 h(x_k)$ is positive definite. At the current iteration point x_k, the function h is represented using the quadratic approximation $q^{(k)}$ as

$$h(x_k + s) \approx h(x_k) + \nabla h(x_k)^\top s + \frac{1}{2} s^\top \nabla^2 h(x_k) s,$$

where $s = x - x_k$. In Newton's method, the search direction is determined as follows

$$d_k = -\left[\nabla^2 h(x_k)\right]^{-1} \nabla h(x_k).$$

Where $[\nabla^2 h(x_k)]^{-1}$ is the inverse of the Hessian. Thus, the Newton method can be defined by

$$x_{k+1} = x_k - \alpha_k \left[\nabla^2 h(x_k)\right]^{-1} \nabla h(x_k),\ k = 0, 1, 2, \ldots \tag{1.3.1}$$

In Newton method, d_k is a descent direction if and only if the Hessian matrix $\nabla^2 h(x_k)$ is positive definite.

Algorithm 2: Newton algorithm

Step 0: Provide x_0 and set $k = 0$.

Step 1: Compute $d_k = -[\nabla^2 h(x_k)]^{-1} \nabla h(x_k)$.

Step 3: Set $x_{k+1} = x_k + d_k$.

Step 4: set $k = k + 1$ and and return to **Step 1**.

Gradient method

In 1847, Cauchy introduced the gradient method, which remains one of the simplest and most fundamental techniques for unconstrained optimization [15]. The method determines the descent direction as $d_k = -\nabla h(x_k)$ which makes it a gradient-based descent approach.

At any given point x_k, the negative gradient direction is the most effective for locating the minimum of the objective function h. For this reason, it is often called the steepest descent method.

Algorithm 3: The gradient algorithm

Step 0: **(Initialization)**

- Choose an initial point x_0, and let ϵ be the precision.
- Set $k = 0$.

Step 1:
- If $\|\nabla h(x_k)\| \leq \epsilon$, then STOP. The solution is $x^* = x_k$.
- Otherwise, let $d_k = -\nabla h(x_k)$ and go to **Step 2**.

Step 2:
- Compute α_k using a line search method.
- Compute the new iterate $x_{k+1} = x_k + \alpha_k d_k$
- Set $k = k + 1$ and return to **Step 1**.

1.3.2 Line search methods

Exact line search Methods

Since the objective is to minimize a function h, it is natural to determine the step-length α_k by addressing the following one-dimensional problem

$$\min_{\alpha>0} \phi_k(\alpha) = \min_{\alpha>0} h(x_k + \alpha d_k).$$

This method is called the Cauchy rule, and the step-length derived from it is termed the Cauchy step or optimal step. In specific cases, it might be more suitable to identify the smallest stationary point of ϕ

$$\alpha_k = \inf\{\alpha > 0 : \phi'(\alpha) = 0,\ \phi(\alpha) < \phi(0)\}.$$

This is known as Cauchy's rule, and the corresponding step is called Cauchy's step. These methods are often collectively referred to as exact line search methods.

Remark 1.3.1 *These rules are generally applied in specific scenarios, such as when ϕ is quadratic, enabling the exact determination of the line search solution within a finite number of iterations.*

- **Advantages and disadvantages of exact line searches:**

  The main advantage of exact line searches lies in their capability to compute the optimal $\alpha_k = \alpha^*$, where

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

  In other words, when $\alpha_k = \alpha^*$, the function h achieves the most effective reduction from the point x_k to $x_k + \alpha_k d_k = x_k + \alpha^* d_k$. However, these techniques are time-intensive, demand substantial memory, and incur high computational costs.

Inexact line search Methods

We analyse a common situation in the application of line search techniques within multidimensional optimization method. At iteration k, the current point is $x_k \in \mathbb{R}^n$ and the search

direction $d_k \in \mathbb{R}^n$, serves as a descent direction for the objective function $h : \mathbb{R}^n \to \mathbb{R}$, meeting the condition

$$\nabla h(x_k)^\top d_k < 0.$$

The objective is to substantially reduce the function's value by selecting an appropriate step-length α_k along the direction d_k. Prominent researchers, including Armijo, Goldstein, Wolfe, Al-Baali, and Fletcher, have introduced various rules to achieve this goal. Below, we present the key tests associated with these rules.

The Armijo rule (1966)

Let $h : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function and let $x_k \in \mathbb{R}^n$ and $d_k \in \mathbb{R}^n$ be a descent direction, i.e. $\nabla h(x)^\top d < 0$.

$\alpha_k > 0$ satisfies the Armijo condition, if

$$h(x_k + \alpha_k d_k) \leq h(x_k) + \delta \alpha_k \nabla h(x_k)^\top d_k, \quad \delta \in]0,1[.$$

We can express

$$\phi_k(\alpha) = h(x_k + \alpha d_k).$$

We then derive

$$\begin{aligned} \phi_k(0) &= h(x_k), \\ \phi_k'(\alpha) &= \nabla h(x_k + \alpha d_k)^\top d_k, \end{aligned}$$

and

$$\phi_k'(0) = \nabla h(x_k)^\top d_k < 0.$$

Armijo test:

- If $\phi_k(\alpha) \leq \phi_k(0) + \delta \alpha \phi_k'(0)$, then α is considered suitable.
- If $\phi_k(\alpha) > \phi_k(0) + \delta \alpha \phi_k'(0)$, then α is too large.

Remark 1.3.2 *In general, the Armijo rule ensures that α_k is sufficiently large, as a very small step-length might hinder the algorithm's progress and lead to premature convergence at a non-stationary point.*

Algorithm 4: The Armijo rule.

- **Step 0 :** (**Initialization**) $a_0 = 0$, $b_0 > a_0$, $\alpha_0 > 0$, $\delta \in (0,1)$, and set $k = 0$.
- **Step 1 :** -If $\phi(\alpha_k) \leq \phi(0) + \delta\alpha_k\phi'(0)$ then, Terminate and set $\alpha^* = \alpha_k$.
  -else $\phi(\alpha_k) > \phi(0) + \delta\alpha_k\phi'(0)$ then, set $a_{k+1} = a_k$, $b_{k+1} = \alpha_k$.
- **Step 2:** -If $b_{k+1} = 0$ determine $\alpha_{k+1} \in]a_{k+1}, +\infty[$. Otherwise determine $\alpha_{k+1} \in]a_{k+1}, b_{k+1}[$.
  - Set $k = k+1$ and return to **Step 1**.

Theorem 1.3.2 *Let $\phi : \mathbb{R}_+^* \to \mathbb{R}$, be a function defined by $\phi_k(\alpha) = h(x_k + \alpha d_k)$. Assume that ϕ_k is bounded and continuous below, d_k is a descent direction at x_k, and $\delta \in]0,1[$. Under these conditions, there exists a non-empty set of steps that satisfy the Armijo rule.*

Proof. see [30] ■

The Goldstein-Price rule (1969)

By introducing a second inequality to the Armijo rule, we derive the Goldstein-Price rule. Here, δ and μ are two constants satisfying $0 < \delta < \mu < 1$. The Goldstein-Price rule consists of the following two inequalities

$$h(x_k + \alpha_k d_k) \leq h(x_k) + \delta\alpha_k \nabla h(x_k)^\top d_k, \quad \delta \in]0,1[,$$
$$h(x_k + \alpha_k d_k) \geq h(x_k) + \mu\alpha_k \nabla h(x_k)^\top d_k, \quad \mu \in]\delta,1[.$$

These conditions can be rewritten as

$$\phi_k(\alpha) \leq \phi_k(0) + \delta\alpha\phi_k'(0),$$
$$\phi_k(\alpha) \geq \phi_k(0) + \mu\alpha\phi_k'(0).$$

The test of Goldstein-Price:

-If $\phi_k(0) + \mu\alpha\phi_k'(0) \leq \phi_k(\alpha) \leq \phi_k(0) + \delta\alpha\phi_k'(0)$, the step-length α is appropriate.

-If $\phi_k(\alpha) > \phi_k(0) + \delta\alpha\phi_k'(0)$, the step-length α is excessively large.

-If $\phi_k(\alpha) < \phi_k(0) + \mu\alpha\phi_k'(0)$, the step-length α is insufficiently large.

Algorithm 5: Goldstein-Price's rule

- **Step 0 :(Initialization)**
  $a_0 = 0, b_0 > 0$, select $\alpha_0 \in [a_0, b_0]$, $\delta \in]0, 1[$, $\mu \in]\delta, 1[$ and set $k = 0$.
- **Step 1 :** -If $\phi(\alpha_k) \leq \phi(0) + \delta\alpha_k\phi'(0)$, then go to **Step 2**.
  - Otherwise, put $a_{k+1} = a_k$, $b_{k+1} = \alpha_k$ and proceed to **Step 3**.
- **Step 2 :**
  -If $\phi(\alpha_k) \geq \phi(0) + \mu\alpha_k\phi'(0)$ then, **Stop** and put $\alpha^* = \alpha_k$.
  -Else, set $a_{k+1} = \alpha_k$, $b_{k+1} = b_k$.
- **Step 3 :**
  -If $b_{k+1} = 0$ determine $\alpha_{k+1} \in]a_{k+1}, +\infty[$. Otherwise determine $\alpha_{k+1} \in]a_{k+1}, b_{k+1}[$.
  - Set $k = k + 1$ and return to **Step 2**.

Theorem 1.3.3 *Let $\phi : \mathbb{R}_+^* \to \mathbb{R}$, be a function defined by $\phi_k(\alpha) = h(x_k + \alpha d_k)$. if ϕ_k is lower bounded and continuous, d_k is a descent direction at x_k, $\delta \in (0, 1)$ and $\mu \in (\delta, 1)$. Under these conditions, there exists a non-empty set of steps that satisfy the Goldstein-Price rule.*

Proof. See [30] for a detailed proof. ■

Wolfe's rule

In Wolfe rule, the step-length α_k is determined to satisfy the following two inequalities, referred to as the Wolfe conditions, where $\delta, \sigma \in \mathbb{R}$ such that $0 < \delta < \sigma < 1$:

$$h(x_k + \alpha_k d_k) \leq h(x_k) + \delta\alpha_k \nabla h(x_k)^\top d_k, \tag{W_1}$$

$$\nabla h(x_k + \alpha_k d_k)^\top d_k \geq \sigma \nabla h(x_k)^\top d_k, \tag{W_2}$$

These conditions can be reformulated as

$$\phi_k(\alpha) \leq \phi_k(0) + \delta\alpha\phi_k'(0),$$
$$\phi_k'(\alpha) \geq \sigma\phi_k'(0).$$

Wolfe test:

-If $\phi_k(\alpha) \leq \phi_k(0) + \delta\alpha\phi_k'(0)$ and $\phi_k'(\alpha) \geq \sigma\phi_k'(0)$, the step-length α is appropriate.

-If $\phi_k(\alpha) > \phi_k(0) + \delta\alpha\phi_k'(0)$, the step-length α is too large.

-If $\phi_k'(\alpha) < \sigma\phi_k'(0)$, the step-length α is too small.

Algorithm 6: Wolfe's rule

- **Step 0 : (Initialization)** - $\alpha_0 > 0, \delta \in (0,1), \sigma \in (\rho,1)$, $a_0 = 0, b_0 > a_0$, and set $k = 0$.
- **Step 1 :**

  -If $\phi(\alpha_k) \leq \phi(0) + \delta\alpha_k\phi'(0)$, proceed to **Step 2.**

  -Else, set $b_{k+1} = \alpha_k, a_{k+1} = a_k$, proceed to **Step 3**.
- **Step 2 :**

  -If $\phi'(\alpha_k) \geq \sigma\phi'(0)$ then, **Stop** and put $\alpha^* = \alpha_k$.

  - Otherwise, set $a_{k+1} = \alpha_k, b_{k+1} = b_k$.
- **Step 3 :**

  -If $b_{k+1} = 0$ determine $\alpha_{k+1} \in]a_{k+1}, +\infty[$. Otherwise determine $\alpha_{k+1} \in]a_{k+1}, b_{k+1}[$.

  - Set $k = k+1$ and return to. **Step 2**.

The strong Wolfe rule (1971)

As the name implies, the strong Wolfe conditions are stronger than the standard Wolfe conditions. They are commonly used in theoretical contexts and play a crucial role in significant convergence results. The step-length α_k satisfies the strong Wolfe conditions, which means that α_k meets the following

$$h(x_k + \alpha_k d_k) \leq h(x_k) + \delta\alpha_k \nabla h(x_k)^\top d_k, \tag{SW_1}$$

$$\left|\nabla h(x_k + \alpha_k d_k)^\top d_k\right| \leq -\sigma \nabla h(x_k)^\top d_k, \tag{SW_2}$$

Equivalently,

$$\begin{aligned}\phi_k(\alpha) &\leq \phi_k(0) + \delta\alpha\varphi'_k(0),\\ \left|\phi'_k(\alpha)\right| &\leq -\sigma\phi'_k(0).\end{aligned}$$

Where $0 < \delta < \sigma < 1$.

Remark 1.3.3 - *The strong Wolfe conditions imply the weak Wolfe conditions*
-(W_1) and (SW_1) represent the same condition, and similarly, (W_2) and (SW_2) are equivalent because

$$\begin{aligned}\left|\nabla h(x_k+\alpha_k d_k)^\top d_k\right| \leq -\sigma\nabla h(x_k)^\top d_k & \\ \Leftrightarrow \sigma\nabla h(x_k)^\top d_k \leq \nabla h(x_k+\alpha_k d_k)^\top d_k \leq -\sigma\nabla h(x_k)^\top d_k & \\ \Rightarrow \sigma\nabla h(x_k)^\top d_k \leq \nabla h(x_k+\alpha_k d_k)^\top d_k &\end{aligned}$$

Theorem 1.3.4 *Let $\phi : \mathbb{R}^*_+ \to \mathbb{R}$, be a differentiable and bounded-below function, defined as $\phi_k(\alpha) = h(x_k + \alpha d_k)$. Assume that ϕ_k is bounded and continuous below, d_k is a descent direction at x_k, and $\delta \in]0,1[$. Under these conditions, the set of steps satisfying the strong Wolfe criterion is non-empty.*

Proof. See [30] for a detailed proof. ■

Algorithm 7: The strong Wolfe rule

- **Step 0 : (Initialization)** $\alpha_0 > 0, \delta \in (0,1)$, $\sigma \in (\delta, 1)$, $a_0 = 0, b_0 > a_0$ and set $k = 0$.
- **Step 1 :**
  -If $\phi(\alpha_k) \leq \phi(0) + \delta\alpha_k\phi'(0)$, and proceed to **Step 2.**
  -Else, set $b_{k+1} = \alpha_k, a_{k+1} = a_k$, and proceed to **Step 3**.
- **Step 2 :**
  -If $\left|\phi'(\alpha_k)\right| \leq \sigma\phi'(0)$ then, **Stop** and put $\alpha^* = \alpha_k$.
  - Otherwise, set $a_{k+1} = \alpha_k, b_{k+1} = b_k$.
- **Step 3 :**
  -If $b_{k+1} = 0$ determine $\alpha_{k+1} \in]a_{k+1}, +\infty[$. Otherwise determine $\alpha_{k+1} \in]a_{k+1}, b_{k+1}[$.
  - Set $k = k+1$ and return to **Step 2**.

1.3.3 The linear conjugate gradient methods

The main idea of this method is to construct a sequence of descent directions that are mutually conjugate with respect to the quadratic function, in order to solve the following unconstrained optimization problem

$$\{\min q(x), \quad x \in \mathbb{R}^n\}$$

where $q(x) = \frac{1}{2}x^\top Ax - b^\top x + c$, is a strictly convex quadratic function.

- $A \in \mathbb{M}_{n\times n}$ is a symmetric positive definite matrix.
- b is a vector in $\mathbb{R}^n$.
- $c \in \mathbb{R}$ is a constant.

The conjugate gradient method is an iterative method that generates a sequence $\{x_k\}$, starting with an initial point x_0 following the procedure

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

where $\{d_0, d_1, \ldots, d_n\}$ is a set of descent directions that have the property of being A-conjugate. The descent direction d_k is given by the following recurrence formula

$$d_k = \begin{cases} -\nabla q(x_0), & \text{if } k = 0, \\ -\nabla q(x_k) + \beta_k d_{k-1}, & \text{if } k \geq 1. \end{cases}$$

Here, the gradient of q at point x_k is denoted by $g_k = \nabla q(x_k)$.

The step-length $\alpha_k \in \mathbb{R}^+$ is determined via a line search technique. The coefficient $\beta_k \in \mathbb{R}$ is chosen such that the directions d_k are A-conjugate. The algorithm below summarizes the main steps of the linear conjugate gradient methods.

Algorithm 8: The linear conjugate gradient algorithm

Step 0: **(Initialization)** Let x_0 be the starting point, $g_0 = \nabla q(x_0) = Ax_0 - b, d_0 = -g_0$ and let ϵ be the termination criterion. Set $k = 0$.

Step 1: If $\|g_k\| = 0$: STOP ($x^* = x_k$). Otherwise, go to **Step 2**.

Step 2: Compute $\alpha_k = \frac{-d_k^\top g_k}{d_k^\top A d_k}$, set $x_{k+1} = x_k + \alpha_k d_k$and, set $k = k + 1$.

Step 2: Determine $d_k = -g_k + \beta_k d_{k-1}$, *where* $\beta_k = \frac{g_k^\top A d_{k-1}}{d_{k-1}^\top A d_{k-1}}$.
- Go back to **Step 1**.

1.3.4 The nonlinear conjugate gradient methods

The conjugate gradient method was extended to nonlinear cases by Fletcher and Reeves in 1964 to minimize general smooth functions $h : \mathbb{R}^n \to \mathbb{R}$ and was further developed by Polak, Ribiére, and Polyak in 1969. More recently, new nonlinear conjugate gradient methods have been developed for unconstrained optimization problems. These methods generate a sequence of points $\{x_k\}$, where the next point is computed as

$$x_{k+1} = x_k + \alpha_k d_k, \tag{1.3.2}$$

where the step-length α_k is determined via a line search. The search direction d_k is given by

$$d_k = \begin{cases} -g_k & \text{if} \quad k = 0, \\ -g_k + \beta_k d_{k-1} & \text{if} \quad k \geq 1, \end{cases} \tag{1.3.3}$$

where $g_k = \nabla h(x_k)$ is the gradient of h at x_k and β_k is a scalar. The primary distinction between different conjugate gradient methods lies in the choice of the parameter β_k, which will be discussed in detail in the next subsection.

Algorithm 9: The nonlinear conjugate gradient algorithm

Step 0: **(Initialization)** Select $x_0 \in \mathbb{R}^n$ and let ϵ be the termination criterion , compute $h(x_0)$ and $g(x_0)$. Let $d_0 = -g_0$.

Step 1: If $\|g_k\|_\infty \leq \epsilon$, then stop. Otherwise, go to the next step.

Step 2: Determine the step-length α_k using a line search process.
- Set $x_{k+1} = x_k + \alpha_k d_k$ and $k = k+1$.

Step 3: Compute β_k.

Step 4: $d_k = -g_k + \beta_k d_{k-1}$ and go to the **Step 1**

Standard conjugate gradient methods

In the previous decades, different CG methods have been suggested including: HS (Hestenes and Stiefel, 1952 [34]), FR (Fletcher and Reeves, 1964[27]), PRP (Polyak, 1969; Polak and Ribière, 1969[53]), CD (conjugate descent, Fletcher, 1987[26]), LS (Liu and Storey, 1992 [45]), and DY (Dai and Yuan, 2000 [19]), whose formulas are given as

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

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

where $\|.\|$ is the Euclidean norm in $\mathbb{R}^n$ and $y_k = g_k - g_{k-1}$. In the case of a strictly convex function with an exact line search all the variants mentioned above are equivalent. However, they respond differently when applied to nonlinear objective functions with inexact line searches.

Convergence of nonlinear conjugate gradient methods

Nonlinear conjugate gradient methods are widely studied for their convergence properties. The first convergence result for the nonlinear conjugate gradient method with inexact line searches was established by Al-Baali [7]. In general, the convergence analysis of conjugate gradient algorithms relies on the CG assumptions and certain standard hypotheses regarding the line search.

Assumption 1 *The level set $\mathcal{S} = \{x \in \mathbb{R}^n : h(x) \leq h(x_0)\}$ is bounded, i.e. there exists a constant $K > 0$ such that $\|x\| \leq K, \quad \forall x \in \mathcal{S}$.*

Assumption 2 *In a close neighbourhood $\mathcal{N}$ of $\mathcal{S}$, the objective function h is continuously differentiable and its gradient g is Lipschitz continuous, i.e. there exists a positive constant L such that*

$$\|g(x) - g(y)\| \leq L\|x - y\|, \forall x, y \in \mathcal{N}.$$

Note that Assumptions 1 and 2 imply that there exists a positive constant r such that

$$\|g(x)\| \leq r, \forall x \in \mathcal{N}. \tag{1.3.4}$$

The following theorem ensures the convergence of all variants of the nonlinear conjugate gradient method with Wolfe inexact line searches for arbitrary functions.

Theorem 1.3.5 *Assume that Assumptions 1 and 2 hold and consider any CG method that follows the form (1.3.2) where d_k is a descent direction and α_k satisfies the strong Wolfe line search. If the Zoutendijk condition*

$$\sum_{k \geq 0} \frac{(g_k^\top d_k)^2}{\|d_k\|^2} < +\infty, \tag{1.3.5}$$

holds, then

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

Proof. For the proof, see [68] ■

Remark 1.3.4 *In the fourth and fifth chapters, Theorem 1.3.5, Assumption 1, and 2 will be utilized to prove the global convergence of the new conjugate gradient methods.*

1.4 Interior Point Methods

As the name implies, interior point methods carefully avoid the boundary of the feasible set, thereby bypassing the combinatorial complexity. These methods were originally developed in the 1960s to address nonlinear constrained optimization problems. However, their use in linear programming initially received little attention due to the near-total dominance of the simplex method at the time. The situation changed with the introduction of Karmarkar's algorithm for linear programming in 1984 [37], sparking a revolution in optimization theory. Interior point methods became competitive with the simplex method [22], particularly for large-scale problems involving tens of thousands of variables or constraints. These methods can be classified into three main categories, described below, in all these methods, the non-negativity constraint $x \geq 0$ is replaced with either an ellipsoid or a barrier function.

1.4.1 Potential reduction methods

Linear programming models are widely used in operations research to solve various real-world problems, such as agriculture, telecommunications, transportation, profit maximization, and cost reduction. These models are also used by military forces globally. Solving large linear programming problems was difficult until Karmarkar's [37] efficient interior-point method was developed. This breakthrough allowed the solution of problems with millions of variables and equations. Karmarkar's algorithm introduced a potential function that ensures convergence and polynomial complexity, making it the first interior-point method to successfully compete with the simplex method, especially for large-scale problems.

1.4.2 Central path methods

These methods emerged alongside potential reduction methods and were developed in the early 1990s. They possess strong theoretical properties, such as polynomial complexity and

rapid superlinear convergence. Trajectory central (TC) methods confine the iterates to a neighborhood around the central path, which consists of an arc of perfectly feasible points.

1.4.3 Barrier methods

Barrier methods address the original problem (P_2) by solving a sequence of unconstrained optimization subproblems. The core concept is to choose a penalty function $B(x)$ and a constant ρ so that the optimal solution $x(\rho)$ of the modified problem eventually serves as an optimal solution to the original problem. Penalty methods often assume the absence of equality constraints or that any equality constraints have been transformed into inequality constraints, although this may unnecessarily complicate the problem and potentially violate the linear independence condition. In barrier methods, the initial point x_0 is required to lie within the interior of the feasible set of (P_2).

Numerous interior point methods of the logarithmic barrier type have been proposed in the literature. Crouzeix and Merikhi [18] were the first to develop a logarithmic barrier algorithm based on majorant functions specifically designed for semidefinite programming. Fellahi and Merikhi [24] introduced novel majorant functions for nonlinear programming. More recently, Leulmi et al. focused on constructing minorant functions for semidefinite programming [42], while [41] and [43] investigated the use of a logarithmic barrier algorithm via minorant functions in linear and nonlinear programming, respectively.

Definition 1.4.1 *(**Barrier function**).*
Let $D \subset \mathbb{R}^n$ be a domain. A function B defined on the interior of D, denoted as $int(D)$, is called a barrier function associated with D if

- *B is continuous,*
- *$B(x) \geq 0$ for all $x \in int(D)$,*
- *$\lim_{x \to x^*} B(x) = +\infty$ as $x \to x^*, x^* \in Fr(D)$,*

where $Fr(D)$ denotes the boundary of D.

The most commonly used barrier functions are the logarithmic and inverse functions.

CHAPTER

2 Theoretical and Numerical Results for Nonlinear Optimization Problems

This chapter focuses on the theoretical and numerical analysis of a nonlinear convex optimization problem constrained by equality conditions. Our aim is to introduce a novel approach for solving a broad class of nonlinear optimization problems using a logarithmic barrier interior point method, incorporating a vector $\rho \in \mathbb{R}^n_+$ as a penalty term that utilizes newly defined minorant functions. First, we calculate the search direction using Newton's method. Then, we propose a new alternative for determining the step-length, which simplifies and accelerates the computation. Finally, through numerical experiments on various test problems, we illustrate the enhanced performance of our minorant functions compared to wolfe line search method.

2.1 Preliminary

We introduce the following inequality pertaining to a statistical series $\{u_1, u_2, ..., u_n\}$ consisting of n real numbers. Wolkowicz et al.[63] demonstrate that

$$\begin{aligned} \bar{u} - \sigma_u \sqrt{n-1} &\leq \min_i u_i \leq \bar{u} - \frac{\sigma_u}{\sqrt{n-1}}, \\ \bar{u} + \frac{\sigma_u}{\sqrt{n-1}} &\leq \max_i u_i \leq \bar{u} + \sigma_u \sqrt{n-1}. \end{aligned} \tag{2.1.1}$$

where the mean $\bar{u}$ and the standard deviation are defined respectively by

$$\bar{u} = \frac{1}{n}\sum_{i=1}^{n} u_i \quad \text{and} \quad \sigma_u^2 = \frac{1}{n}\sum_{i=1}^{n} u_i^2 - \bar{u}^2 = \frac{1}{n}\sum_{i=1}^{n}(u_i - \bar{u})^2,$$

Furthermore, Crouzeix and Merikhi [18] presented the following useful inequalities concerning a statistical series where $u_i > 0$ for all $i = \overline{1,n}$

$$n\ln\left(\bar{u}-\sigma_u\sqrt{n-1}\right) \le A \le \sum_{i=1}^{n}\ln(u_i) \le B \le n\ln(\bar{u}), \tag{2.1.2}$$

where

$$A = (n-1)\ln\left(\bar{u}+\frac{\sigma_u}{\sqrt{n-1}}\right)+\ln\left(\bar{u}-\sigma_u\sqrt{n-1}\right),$$
$$B = \ln\left(\bar{u}+\sigma_u\sqrt{n-1}\right)+(n-1)\ln\left(\bar{u}-\frac{\sigma_u}{\sqrt{n-1}}\right).$$

2.2 Original Problem Formulation and its Perturbed Version

Consider the following nonlinear optimization problem with constraints

$$\{\min h(x): \quad x \in \mathscr{S}\}, \tag{P}$$

where h is twice continuously differentiable and convex on $\mathscr{S}$. The sets of feasible and strictly feasible solutions to (P) are defined as follows

$$\mathscr{S} = \{x \in \mathbb{R}^n : x \ge 0, Ax = c\}, \tag{2.2.3}$$
$$\mathscr{S}_0 = \{x \in \mathbb{R}^n : x > 0, Ax = c\}, \tag{2.2.4}$$

such that $c \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m\times n}$ matrix.

Prior to presenting the perturbed version of (P), we state the following mild assumptions

1. $A \in M^{m\times n}$ is a full-rank matrix, $c \in \mathbb{R}^m (m < n)$.
2. The set of optimal solutions for problem (P) is both nonempty and bounded.
3. There exists $x_0 > 0$ satisfying $Ax_0 = c$.

According to the optimality conditions, x^* is an optimal solution of (P) if and only if there exists $u^* \in \mathbb{R}^m$ and $v^* \in \mathbb{R}^n_+$

$$\nabla h(x^*) + A^t u^* - v^* = 0, \quad Ax^* = c, \quad \langle v^*, x^*\rangle = 0. \tag{2.2.5}$$

2.2.1 The perturbed associated problem

In what follows, we reformulate the nonlinear problem (P) as a perturbed problem using a barrier function, where the penalty term ρ is defined as a vector in $\mathbb{R}^n$ and each ρ_i is strictly positive for $i = \overline{1,n}$.

We introduce the function $\varphi : \mathbb{R}^n_+ \times \mathbb{R}^n \to \mathbb{R}^n \cup \{+\infty\}$ as follows

$$\varphi(\rho, x) = \begin{cases} h(x) + \vartheta(\rho, x) & \text{if} \quad x, \rho \geq 0 \quad \text{and} \quad Ax = c. \\ +\infty & \text{otherwise,} \end{cases}$$

where $\vartheta : \mathbb{R}^n_+ \times \mathbb{R}^n \to \mathbb{R}^n \cup \{+\infty\}$ is a function, given by

$$\vartheta(\rho, x) = \begin{cases} \sum_{i=1}^n \rho_i \ln(\rho_i) - \sum_{i=1}^n \rho_i \ln(x_i) & \text{if} \quad x, \rho > 0, \\ 0 & \text{if} \quad x \geq 0 \text{ and } \rho = 0, \\ +\infty & \text{otherwise.} \end{cases}$$

Both functions are convex, lower semicontinuous, and proper. We now define the convex function Ω as

$$\Omega(\rho) = \inf_x \{\varphi_\rho(x) = \varphi(\rho, x) : x \in \mathbb{R}^n\}. \tag{P_ρ}$$

Remark 2.2.1 *We observe that the two problems (P) and (P_ρ) coincide as $\|\rho\| \to 0$, which indicates that the optimal solution of problem (P_ρ) is an approximated solution of problem (P) when $\|\rho\| \to 0$.*

2.3 Theoretical Concepts

The lemma below establishes the existence and uniqueness of the optimal solution for the perturbed problem (P_ρ).

Lemma 2.3.1 *The problem (P_ρ) admits an optimal solution if and only if the recession cone C_d of φ_ρ is reduced to the origin. i.e.*

$$C_d(\varphi_\rho) = \{d \in \mathbb{R}^n : (\varphi_\rho)_\infty(d) \leq 0, \;\; Ad = 0, d \geq 0\} = \{0\}. \tag{2.3.6}$$

where $(\varphi_\rho)_\infty(d)$ *is the asymptotic function of* φ_ρ*, represented as*

$$(\varphi_\rho)_\infty(d) = \lim_{\alpha\to\infty} \frac{\varphi_\rho(x+\alpha d)-\varphi_\rho(x)}{\alpha}.$$

Proof.

Under Assumption 4, the problem (P) has an optimal solution, implying that the recession cone C_d of the function h is reduced to the origin i.e. $C_d(h) = \{0\}$.

$$\begin{aligned}(\varphi_\rho)_\infty(d) &= \lim_{\alpha\to\infty} \frac{\varphi_\rho(x+\alpha d)-\varphi_\rho(x)}{\alpha},\\ &= \lim_{\alpha\to\infty} \frac{h(x+\alpha d)-h(x)}{\alpha} - \lim_{\alpha\to\infty} \frac{\sum_{i=1}^n \rho_i(\ln(x_i+\alpha d_i)-\ln(x_i))}{\alpha},\\ &= \lim_{\alpha\to\infty} \frac{h(x+\alpha d)-h(x)}{\alpha},\end{aligned}$$

hence,

$$(\varphi_\rho)_\infty(d) = \begin{cases}(h)_\infty(d) & \text{if} \quad Ax = c, d \geq 0,\\ +\infty & \text{otherwise.}\end{cases}$$

which means that $C_d(\varphi_\rho) = C_d(h) = \{0\}$. ■

Considering that φ_ρ is strictly convex, we conclude that the perturbed problem (P_ρ) has a unique optimal solution $x(\rho)$ within its feasible set $\mathscr{S}_0$.

Lemma 2.3.2 *We suppose that* $x(\rho)$ *is the optimal solutions of the perturbed problem* (P_ρ)*, while* x^* *is the optimal solution of the original problem* (P)*.*

$$\text{As } \|\rho\| \to 0, \text{ we have } x(\rho) \to x^*$$

Proof.

Based on the necessary and sufficient optimality conditions, $\exists\, \lambda(\rho) \in \mathbb{R}^m$ such that

$$\nabla h(x(\rho)) - X_\rho^{-1}\rho + A^\top \lambda(\rho) = 0, \quad Ax(\rho) = c, \tag{2.3.7}$$

where X is a diagonal matrix with $X_{ii} = x_i, i = 1, ..., n$.

Remark 2.3.1 *In the rest of the paper we take* $x(\rho) = x_\rho$ *and* $\lambda(\rho) = \lambda_\rho$

The functions x_ρ and λ_ρ are differentiable on $\mathbb{R}^n_+$. Moreover, the pair (x_ρ, λ_ρ) must satisfy the following equation.

$$F(x_\rho, \lambda_\rho) = \begin{pmatrix} \nabla h(x_\rho) - X_\rho^{-1}\rho + A^\top \lambda_\rho \\ Ax_\rho - c \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{2.3.8}$$

Applying the implicit function theorem, we obtain

$$\begin{pmatrix} \nabla^2 h(x_\rho) + PX_\rho^{-2} & A^\top \\ A & 0 \end{pmatrix} \begin{pmatrix} \nabla x_\rho \\ \nabla \lambda_\rho \end{pmatrix} = \begin{pmatrix} X_\rho^{-1} \\ 0 \end{pmatrix}, \tag{2.3.9}$$

with P is a diagonal matrix, $P_{ii} = \rho_i$, for every $i = 1, ..., n$, and

$$\nabla x_\rho = \begin{pmatrix} \frac{\partial x_1}{\partial \rho_1} & \frac{\partial x_1}{\partial \rho_2} & \cdots & \frac{\partial x_1}{\partial \rho_n} \\ \frac{\partial x_2}{\partial \rho_1} & \frac{\partial x_2}{\partial \rho_2} & \cdots & \frac{\partial x_2}{\partial \rho_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial x_n}{\partial \rho_1} & \frac{\partial x_n}{\partial \rho_2} & \cdots & \frac{\partial x_n}{\partial \rho_n} \end{pmatrix}, \quad \nabla \lambda_\rho = \begin{pmatrix} \frac{\partial \lambda_1}{\partial \rho_1} & \frac{\partial \lambda_1}{\partial \rho_2} & \cdots & \frac{\partial \lambda_1}{\partial \rho_n} \\ \frac{\partial \lambda_2}{\partial \rho_1} & \frac{\partial \lambda_2}{\partial \rho_2} & \cdots & \frac{\partial \lambda_2}{\partial \rho_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \lambda_n}{\partial \rho_1} & \frac{\partial \lambda_n}{\partial \rho_2} & \cdots & \frac{\partial \lambda_n}{\partial \rho_n} \end{pmatrix}.$$

From (P_ρ), the gradient of the differentiable function Ω, defined as

$$\nabla \Omega(\rho) = (\nabla x_\rho)^\top (\nabla h(x_\rho) - X_\rho^{-1}\rho) + (e + l_1 - l_2)$$

where $e = (1, 1, ..., 1)^\top$, $l_1 = (\ln(\rho_1), \ln(\rho_2), ..., \ln(\rho_n))^\top$ and $l_2 = (\ln(x_1), \ln(x_2), ..., \ln(x_n))^\top$.

Using (2.3.7) and (2.3.9), we find

$$\begin{aligned} \nabla \Omega(\rho) &= -(\nabla x_\rho)^\top A^\top \lambda_\rho + (e + l_1 - l_2) \\ &= -(A \nabla x_\rho)^\top \lambda_\rho + (e + l_1 - l_2) \\ &= e + l_1 - l_2. \end{aligned}$$

With Ω being convex and for $x_\rho \in \mathscr{S}$, we get

$$h^* = \Omega(0) = \min_x h(x^*).$$

This leads to

$$h^* \leq h(x_\rho) \leq h^* + \sum_{i=1}^{n} \rho_i,$$

indeed, we have the following steps

$$\begin{aligned}\Omega(0) &\geq \Omega(\rho) - \rho^\top \nabla\Omega(\rho)\\ &\geq h(x_\rho) + \sum_{i=1}^{n}\rho_i \ln(\rho_i) - \sum_{i=1}^{n}\rho_i \ln((x_i)_\rho) - \rho^\top(e + l_1 - l_2)\\ &\geq h(x_\rho) + \sum_{i=1}^{n}\rho_i \ln(\rho_i) - \sum_{i=1}^{n}\rho_i \ln((x_i)_\rho)\\ &-\sum_{i=1}^{n}\rho_i - \sum_{i=1}^{n}\rho_i \ln(\rho_i) + \sum_{i=1}^{n}\rho_i \ln((x_i)_\rho)\\ &\geq h(x_\rho) - \sum_{i=1}^{n}\rho_i.\end{aligned}$$

Consequently, we have

$$h^* \leq h(x_\rho) \leq h^* + \sum_{i=1}^{n}\rho_i.$$

The trajectory of x_ρ when $\|\rho\|$ approaches 0:

(i) h **is merely convex:** Let $\|\rho\|_\infty \leq 1$. The following is bounded, non-empty and convex set

$$x_\rho \in \{x : Ax - c = 0, x > 0, h(x) \leq n + h^*\}.$$

The recession cone of this set is reduced to the origin. Furthermore, each accumulation point of x_ρ becomes an optimal solution to (P) only as $\|\rho\|$ approaches to zero.

(ii) h **is strongly convex with a strictly positive parameter δ:** the convergence of x_ρ towards x^* is of order $\frac{1}{2}$, in fact

$$\sum_{i=1}^{n}\rho_i \geq h(x_\rho) - h(x^*) \geq \langle \nabla h(x^*), x_\rho - x^* \rangle + \frac{\delta}{2}\|x_\rho - x^*\|^2,$$

utilizing (2.2.5), we arrive at

$$\sum_{i=1}^{n}\rho_i \geq \langle v^*, x_\rho \rangle + \frac{\delta}{2}\|x_\rho - x^*\|^2 \geq 0.$$

Then we can conclude that

$$\|x_\rho - x^*\| \leq \left(\frac{2}{\delta}\sum_{i=1}^{n}\rho_i\right)^{\frac{1}{2}}.$$

■

2.4 Solution of the Perturbed Problem (P_ρ)

This section is dedicated to finding the numerical solution of our perturbed problem, which is defined as

$$\{\min_x \varphi_\rho(x) : x \in \mathbb{R}^n\} \tag{P_ρ}$$

We begin by calculating the descent direction and utilize a novel minorant function technique to determine the step-length.

2.4.1 The descent direction

In this study, we apply Newton's method, which leads to d being determined by solving the following quadratic convex minimization problem

$$\begin{cases} \min\limits_d \left\{ \frac{1}{2}\langle \nabla^2 \varphi_\rho(x)d, d\rangle + \langle \nabla \varphi_\rho(x), d\rangle \right\}, \\ Ad = 0. \end{cases}$$

Based on the necessary and sufficient conditions for optimality, there exists a vector $v \in \mathbb{R}^m$ such that

$$\begin{cases} \nabla^2 \varphi_\rho(x)d + \nabla \varphi_\rho(x) + A^\top v = 0, \\ Ad = 0, \end{cases}$$

which corresponds to

$$\begin{pmatrix} \nabla^2 h(x) + PX^{-2} & A^\top \\ A & 0 \end{pmatrix} \begin{pmatrix} d \\ v \end{pmatrix} = \begin{pmatrix} X^{-1}\rho - \nabla h(x) \\ 0 \end{pmatrix}.$$

Then, we achieve

$$\begin{pmatrix} d^\top & 0 \end{pmatrix} \begin{pmatrix} \nabla^2 h(x) + PX^{-2} & A^\top \\ A & 0 \end{pmatrix} \begin{pmatrix} d \\ v \end{pmatrix} = \begin{pmatrix} d^\top & 0 \end{pmatrix} \begin{pmatrix} X^{-1}\rho - \nabla h(x) \\ 0 \end{pmatrix},$$

As a result, we obtain the following relationship

$$\langle \nabla^2 h(x)d, d\rangle + \langle \nabla h(x), d\rangle = \langle \rho, X^{-1}d\rangle - \langle PX^{-1}d, X^{-1}d\rangle, \tag{2.4.10}$$

equivalent to

$$\begin{pmatrix} X\nabla^2 h(x)X + P & XA^\top \\ AX & 0 \end{pmatrix} \begin{pmatrix} X^{-1}d \\ v \end{pmatrix} = \begin{pmatrix} \rho - X\nabla h(x) \\ 0 \end{pmatrix}.$$

This linear system can be solved efficiently using Cholesky decomposition.

2.4.2 Step-length determination

Once the direction is computed, we determine α to ensure a significant decrease in the search line function. The next iteration is then updated as $x+\alpha d$. In this part, our goal is to determine the step-length α. To achieve this, we employ a straightforward and effective method that is more efficient than traditional line search methods (e.g., Armijo, Goldstein, Wolfe, Fibonacci, etc.). This approach involves utilizing a minorant function of the line search function.

Consider the following function

$$G_g(\alpha)=\varphi_\rho(x+\alpha d)-\varphi_\rho(x)=h(x+\alpha d)-h(x)-\sum_{i=1}^{n}\rho_i\ln(1+\alpha y_i),$$

where $y=X^{-1}d$ and $\alpha\in]0,\widehat{\alpha}_0[$, such that $\widehat{\alpha}_0=\max\{\alpha:1+\alpha y_i>0\}$.
By using the inequality in (2.1.1), we have

$$\rho_i\leq\max_i\rho_i\leq\bar{\rho}+\sigma_\rho\sqrt{n-1},\ \ \forall i=1,...,n.$$

Setting $\tau=\bar{\rho}+\sigma_\rho\sqrt{n-1}$. For every $\alpha\in]0,\widehat{\alpha}_0[$, we achieve

$$G(\alpha)\geq G_0(\alpha)=\frac{1}{\tau}(h(x+\alpha d)-h(x))-\sum_{i=1}^{n}\ln(1+\alpha y_i). \tag{2.4.11}$$

with $G(\alpha)=\frac{G_g(\alpha)}{\tau}$. It is straightforward to prove that

$$G'(\alpha)=\frac{1}{\tau}\left(\langle\nabla h(x+\alpha d),d\rangle-\sum_{i=1}^{n}\rho_i\frac{y_i}{1+\alpha y_i}\right),$$
$$G''(\alpha)=\frac{1}{\tau}\left(\langle\nabla^2 h(x+\alpha d)d,d\rangle+\sum_{i=1}^{n}\rho_i\frac{y_i^2}{(1+\alpha y_i)^2}\right),$$

and

$$G_0'(\alpha)=\frac{1}{\tau}\left(\langle\nabla h(x+\alpha d),d\rangle-\sum_{i=1}^{n}\frac{y_i}{1+\alpha y_i}\right),$$
$$G_0''(\alpha)=\frac{1}{\tau}\left(\langle\nabla^2 h(x+\alpha d)d,d\rangle+\sum_{i=1}^{n}\frac{y_i^2}{(1+\alpha y_i)^2}\right).$$

G and G_0 meet the requirement for significant decrease, since:
From (2.4.10), we have

(i) $G'(0)+G''(0)=0$ and $G''(0)\geq 0$, it follows that $G'(0)\leq 0$.

(ii) if $y_i\geq 0$, It is evident that $G_0'(0)\leq 0$.

(iii) if $y_i<0$, we have $G_0'(0)+G_0''(0)\leq 0$ and $G_0''(0)\geq 0$. Thus $G_0'(0)\leq 0$.

2.4.3 The new approximating minorant function

In this part, we propose finding an approximation function G_1 for G to address the numerical difficulties in calculating $G'(\alpha)=0$. In our case, this requires that

$$G(0)=G_0(0)=G_1(0)=0,\ \ G_0'(0)=G_1'(0)<0,\ \ G_0''(0)=G_1''(0)>0. \tag{2.4.12}$$

To find G_1 we need to set $x_i=1+\alpha y_i$, $\bar{x}_i=1+\alpha\bar{y}_i$, and $\sigma_x=\alpha\sigma_y$.

Based on (2.1.2), we have

$$\sum_{i=1}^{n}\ln(x_i)\leq B,$$

After a straightforward calculation, we get $G_1(\alpha)\leq G_0(\alpha)$, where

$$G_1(\alpha)=\frac{1}{\tau}(h(x+\alpha d)-h(x))-(n-1)\ln(1+\alpha\gamma)-\ln(1+\alpha\beta),\quad \alpha\in]0,\widehat{\alpha}[$$

with

$$\begin{cases}\gamma &= \bar{y}-\frac{\sigma_y}{\sqrt{n-1}},\\ \beta &= \bar{y}+\sigma_y\sqrt{n-1},\\ \widehat{\alpha} &= \min\{\widehat{\alpha}_0,\max\{\alpha: 1+\alpha\gamma>0\}\}.\end{cases}$$

It is evident that

$$\begin{aligned}G_1'(\alpha) &= \frac{1}{\tau}\langle\nabla h(x+\alpha d),d\rangle-(n-1)\frac{\gamma}{1+\alpha\gamma}-\frac{\beta}{1+\alpha\beta},\\ G_1''(\alpha) &= \frac{1}{\tau}\langle\nabla^2 h(x+\alpha d)d,d\rangle+(n-1)\frac{\gamma^2}{(1+\alpha\gamma)^2}+\frac{\beta^2}{(1+\alpha\beta)^2}.\end{aligned}$$

It is clear that G_1satisfies the conditions in (2.4.12), making the strictly convex function G_1 a good approximation of G_0 in a neighborhood of 0. Furthermore, the unique minimum α^* of G_1 satisfies the following inequality

$$G_1(\alpha^*)\leq G_0(\alpha^*)\leq G(\alpha^*).$$

2.4.4 The auxiliary function ζ_1 of G_1

Lemma 2.4.1 *Let the auxiliary function ζ_1, written as*

$$\zeta_1(\alpha) = n\eta\alpha - (n-1)\ln(1+\alpha\gamma) - \ln(1+\alpha\beta), \tag{2.4.13}$$

where $\gamma = \bar{y} - \frac{\sigma_y}{\sqrt{n-1}}$, and $\beta = \bar{y} + \sigma_y\sqrt{n-1}$.
If the objective function is linear

$$\eta = \frac{c^\top d}{n\tau}.$$

If the objective function is only convex

$$\eta = \frac{(h(x+\bar{\alpha}d) - h(x))}{n\tau\bar{\alpha}}$$

the auxiliary function ζ_1 exhibits the same properties as G_1

Proof.

- **If h is linear**, then, $h(x) = \langle c, x\rangle$ where $c, x \in \mathbb{R}^n$. The auxiliary function can be defined by

  $$\zeta_1(\alpha) = \frac{\langle c, x\rangle}{\tau}\alpha - (n-1)\ln(1+\alpha\gamma) - \ln(1+\alpha\beta),$$

  we can take $\eta = \frac{c^\top d}{n\tau}$, which gives (2.4.13).

  The functions ζ_1 and G_1 coincide. The unique solution to $\zeta_1'(\alpha) = 0$ is the same as the unique root of $G_1'(\alpha) = 0$ that ensures a significant reduction in the function φ_ρ along the direction d.

- **If h only convex**, in this case, the equation $G_1'(\alpha) = 0$ no longer reduces to a quadratic equation. We considered exploring another function than G_1, for which we applied the secant method. Let $\bar{\alpha} \in]0, \widehat{\alpha}[$ for all $\alpha \in]\bar{\alpha}, \widehat{\alpha}[$, then, we have

  $$\frac{\alpha}{\tau\bar{\alpha}}(h(x+\bar{\alpha}d) - h(x)) \leq \frac{1}{\tau}(h(x+\alpha d) - h(x)).$$

  Thus, the auxiliary function is defined by

  $$\zeta_1(\alpha) = \frac{(h(x+\bar{\alpha}d) - h(x))}{\tau\bar{\alpha}}\alpha - (n-1)\ln(1+\alpha\gamma) - \ln(1+\alpha\beta),$$

  we can take $\eta = \frac{(h(x+\bar{\alpha}d) - h(x))}{n\tau\bar{\alpha}}$, which gives (2.4.13).

  After calculating the solution of $\zeta_1'(\alpha) = 0$, we have

(ii) $\alpha^* \leq \bar{\alpha}$ we must take another $\bar{\alpha}$. The computation of $\bar{\alpha}$ is carried out using a dichotomous (see Remark 2.4.1)

(iii) $\alpha^* \geq \bar{\alpha}$, then $\zeta_1(\alpha^*) \leq G_1(\alpha^*) \leq G_0(\alpha^*) \leq G(\alpha^*)$.

The auxiliary function ζ_1 satisfies the following points

- ζ_1 shares the same properties as G_1.
- The unique solution to the equation $\zeta_1'(\alpha) = 0$ defines the minimum α^* of G_1.
- The value α^* ensures a significant reduction of the function φ_ρ along the direction d.

■

The minimization of the auxiliary function

Let

$$\zeta_1(\alpha) = n\eta\alpha - (n-1)\ln(1+\alpha\gamma) - \ln(1+\alpha\beta).$$

we have

$$\zeta_1'(\alpha) = n\eta - (n-1)\frac{\gamma}{1+\alpha\gamma} - \frac{\beta}{1+\alpha\beta},$$
$$\zeta_1''(\alpha) = (n-1)\frac{\gamma^2}{(1+\alpha\gamma)^2} - \frac{\beta^2}{(1+\alpha\beta)^2}.$$

It is worth noting that

$$\begin{cases} \zeta_1(0) & = 0, \\ \zeta_1'(0) & = n(\eta - \bar{y}), \\ \zeta_1''(0) & = n(\bar{y}^2 + \sigma_y^2) = \|y\|^2. \end{cases}$$

We impose that $\zeta_1'(0) \leq 0$ and $\zeta_1''(0) \geq 0$.

To minimize $\zeta_1(\alpha)$, we need to calculate $\zeta_1'(\alpha) = 0$. It is similar to

$$\eta\gamma\beta\alpha^2 + (\eta(\alpha+\beta) - \gamma\beta)\alpha + \eta - \bar{y} = 0.$$

As a result, we find

$$\alpha^* = \begin{cases} \frac{-\bar{y}}{\gamma\beta} & \text{if } \eta = 0, \\ \frac{\bar{y}-\eta}{\eta\beta} & \text{if } \gamma = 0, \\ \frac{\bar{y}-\eta}{\eta\gamma} & \text{if } \beta = 0. \end{cases} \tag{2.4.14}$$

In the case of $\eta\gamma\beta \neq 0$ we find

$$\alpha_1^* = \frac{1}{2}\left(\frac{1}{\eta}-\frac{1}{\gamma}-\frac{1}{\beta}-\sqrt{\Delta}\right),\ \ \alpha_2^* = \frac{1}{2}\left(\frac{1}{\eta}-\frac{1}{\gamma}-\frac{1}{\beta}+\sqrt{\Delta}\right),$$

where

$$\Delta = \frac{1}{\eta^2}+\frac{1}{\gamma^2}+\frac{1}{\beta^2}-\frac{2}{\gamma\beta}-\left(\frac{2n-4}{n\eta}\right)\left(\frac{1}{\gamma}-\frac{1}{\beta}\right).$$

we choose only the root α^* that belongs to the domain of ζ_1, we take $\alpha^* \in]0, \widehat{\alpha}-\epsilon[$, where $\epsilon > 0$.

Remark 2.4.1 *The computation of α^* is performed utilizing a dichotomous method under the conditions that α^* is not in the interval $]0, \widehat{\alpha}-\epsilon[$ and $G'(\alpha^*) > 0$.*

Take $a = 0$, $b = \widehat{\alpha}-\epsilon$.

While $|b-a| > \epsilon$ do

- $c = \frac{a+b}{2}$.

- If $G'(c) < 0$, then $b = c$ else $a = c$.

This method provides a refined approximation of the solution of $G'(\alpha)$ while ensuring compliance with the domain of G.

2.5 The Algorithm

The primary steps of the algorithm for obtaining an optimal solution, x^* , to problem (P) are detailed in the algorithm below.

Algorithm 10: Algorithm of line search function

1 **Step 0:** (Initialization) Select $x_0 \in \mathscr{S}_0$, $X_{ii} = (x_0)_i$ and the parameters $\rho_s > 0$, $\rho \in \mathbb{R}^n_+$, $b \in [0,1]^n \ and \ \epsilon > 0$.

Step 1: Compute d and $y = X^{-1}d$.

Step 2:

- If $\|y\| \leq \epsilon$, then $\Omega(\rho)$ is well approximated. So

  * if $\|\rho\| \leq \rho_s$, then stop (we obtain an accurate approximation of the optimal solution).

  * Otherwise, put $\rho = b \times \rho$ where $b \times \rho = (b_1 \times \rho_1, ..., b_n \times \rho_n)$, and go to the **Step 1.**

- If $\|y\| \geq \epsilon$, then

  * compute η, τ, γ and β.

  * calculate $\alpha^* > 0$ using the equation $\zeta_1'(\alpha) = 0$.

  * compute $x = x + \alpha^* d$ and return to the **Step 1**

2.6 Numerical Experiments

In this section, we present comparative numerical tests based on several examples drawn from [52, 57]. To evaluate the enhanced performance and accuracy of our algorithm, which leverages minorant function, we conduct these tests to compare our new approach with Wolfe line search method. All code was written and executed in MATLAB on a PC with an Intel(R) Core i7-7700HQ (2.80 GHz) and 16 GB of RAM.

In the following tables, we take $\epsilon \leq 10^{-4}$, and

* (itr) represent the iteration numbers.
* (ts) denotes the computational time in seconds.
* (Stm) denotes the strategy of minorant function presented in this paper.
* (LS) refers to the Wolfe line search method .

- **Example 1:** (Erikson's problem [57]).

We examine the following nonlinear problem, where $n = 2m$:

$$\begin{cases} \min h(x) & = \min \sum_{i=1}^{n} x_i \ln\left(\frac{x_i}{a_i}\right) \\ x_i + x_{i+m} & = b, \\ x & \geq 0, \end{cases}$$

where $a_i > 0$ and $b \in \mathbb{R}^m$ are given constants.

The table below summarizes the results obtained for the case $a_i = 2, \forall i = \overline{1,n}$ and $b_i = 4, \forall i = \overline{1,m}$

- **Example 2.** Quadratic case [52]

  We focus on the following quadratic problem, where $n = m + 2$

$$\begin{cases} \min h(x) & = \min \frac{1}{2}\langle x, Qx\rangle \\ Ax & = c, \\ x & \geq 0 \end{cases}$$

(m,n)	Stm		LS	
	itr	ts	itr	ts
$(30,60)$	1	0.0009	3	0.023
$(150,300)$	1	0.0035	4	0.0645
$(300,600)$	2	0.0035	5	3.199
$(500,1000)$	2	0.1112	5	5.324

Table 2.1: Erikson's problem with $a_i = 2, \forall i = \overline{1,n}$ and $b_i = 4, \forall i = \overline{1,m}$

$$Q[i,j] = \begin{cases} 2 & \text{if } i = j = 1 \text{ or } i = j = m \\ 4 & \text{if } i = j \text{ and } i \neq \{1,m\} \\ 2 & \text{if } i = j-1 \text{ or } i = j+1 \\ 0 & \text{otherwise,} \end{cases}$$

and

$$A[i,j] = \begin{cases} 1 & \text{if } i = j \\ 2 & \text{if } i = j-1 \\ 3 & \text{if } i = j-2 \\ 0 & \text{otherwise.} \end{cases}$$

$f_i = 1, \forall i, j = 1,..,n$. We evaluate this example with varying values of n.

(m,n)	Stm		LS	
	itr	ts	itr	ts
30	5	0.0041	26	18.3244
400	3	0.0985	35	60.1003
600	3	9.6544	23	79.0024
1000	5	11.9912	33	91.3358

Table 2.2: Example of Quadratic case (with variable size)

2.6.1 Comments

Table 2.1 presents the performance of the proposed minorant function strategy (Stm) compared to the Wolfe line search (LS) on nonlinear Erikson's minimization problem. The results demonstrate the significant advantage of the proposed approach in terms of both iteration count and computational time. Specifically:

- The Stm method consistently requires fewer iterations (1–2) across all problem sizes, while LS needs 3–5 iterations.
- The computational time using Stm remains extremely low and nearly constant even as the problem size increases, while LS exhibits a sharp increase in time, especially for larger dimensions (e.g., 5.324 seconds vs. 0.1112 for $(500, 1000)$).

Table 2.2 evaluates the same comparison for a structured quadratic optimization problem with equality constraints. The trends observed are similar:

- The Stm method converges within 3 to 5 iterations regardless of the problem size, while LS requires significantly more iterations (23–35).
- Computational time using Stm is orders of magnitude lower for all sizes. For instance, at $n = 1000$, Stm completes in 12 seconds, whereas LS takes over 91 seconds.

These tests clearly highlight the effectiveness of our approach compared to the Wolfe line search method.

CHAPTER 3

A Hybrid CG Algorithm for Nonlinear Unconstrained Optimization with Application in Image Restoration

The conjugate gradient (CG) method is an optimization technique known for its rapid convergence, and has led to significant developments and applications. Numerous variations of CG methods have emerged to enhance computational efficiency in solving unconstrained optimization problems while addressing real-world challenges. In this work, we consider the following unconstrained problem

$$\min_{x\in\mathbb{R}^n} h(x), \tag{3.0.1}$$

where $h:\mathbb{R}^n \longrightarrow \mathbb{R}$ is a continuously differentiable function. The problem (3.0.1) is of interest in many real-world applications involving objective functions which are continuously differentiable [66]. To mention just a few, these are applied to molecular physics [65, 67], statistical modeling [62, 48] and image processing [36]. CG methods are among the most effective methods for solving the problems of type (3.0.1) due to their simplicity and low storage. They take up several forms; their principle is to generate a sequence of points $\{x_k\}_{k\geq 0} \subset \mathbb{R}^n$ starting from an initial point $x_0 \in \mathbb{R}^n$ following the procedure

$$x_{k+1} = x_k + \alpha_k d_k, \tag{3.0.2}$$

where d_k is a descent direction for h at x_k and $\alpha_k \in \mathbb{R}^+$ is a step-length which ensures that x_{k+1} is a feasible point with $h(x_{k+1}) \leq h(x_k)$. The step-length α_k in (3.0.2) is determined by using a line search procedure which ensures that the sufficient decrease conditions are

satisfied at the new point x_{k+1}; typically, it is chosen in such a way that it satisfies the weak Wolfe conditions

$$h(x_k+\alpha_k d_k)-h(x_k)\leq \delta\alpha_k g_k^\top d_k, \qquad (3.0.3)$$
$$\nabla h(x_k+\alpha_k d_k)^\top d_k\geq \sigma g_k^\top d_k,$$

or the strong Wolfe conditions

$$h(x_k+\alpha_k d_k)-h(x_k)\leq \delta\alpha_k g_k^\top d_k,$$
$$|\nabla h(x_k+\alpha_k d_k)^\top d_k|\leq -\sigma g_k^\top d_k, \qquad (3.0.4)$$

where $\delta\in(0,1/2)$, $\sigma\in(\delta,1)$ and $g_k=\nabla h(x_k)$. The search direction d_k is usually defined as

$$d_k=\begin{cases}-g_k & \text{if} \quad k=0,\\ -g_k+\beta_k d_{k-1} & \text{if} \quad k\geq 1,\end{cases}$$

where the parameter β_k is a scalar which determines the different conjugate gradient methods.

This chapter introduces a hybrid conjugate gradient method, referred to as the CR method. The parameter β_k in this method is carefully designed based on the structure of conjugate gradient parameters utilized in certain existing methods. The proposed algorithm satisfies the sufficient descent condition without relying on any line search. Furthermore, it is globally convergent under the usual and strong Wolfe line search assumptions. The method's efficiency is validated through numerical experiments conducted on 100 test problems from [8], as well as image restoration problems. A comparative analysis with existing methods highlights the algorithm's potential and effectiveness.

3.1 The Proposed Method

In this section, a new parameter β_k is introduced, based on a combination of the RMIL (Rivaie-Mustafa-Ismail-Leong) [56] and hSM (hybrid Sulaiman-Mohammed) [62] methods. The new conjugate gradient parameter β_k^{CR} is defined as

$$\beta_k^{CR}=(1-\theta_k)\beta_k^{RMIL}+\theta_k\beta_k^{hSM}, \qquad (3.1.5)$$

where

$$\beta_k^{RMIL} = \frac{g_{k+1}^\top (g_{k+1} - g_k)}{\|d_k\|^2},$$

and

$$\beta_k^{hSM} = \frac{g_{k+1}^\top (g_{k+1} + g_k)}{\|d_k\|^2}.$$

Here $\theta_k \in [0,1]$. To ensure our method generate descent directions which enhance computational efficiency and robustness, we compute the search direction d_k as

$$d_k = \begin{cases} -g_k & \text{if} \quad k = 0, \\ -g_k + \beta_k^{CR}(d_{k-1} - \rho_k g_k) & \text{if} \quad k \geq 1, \end{cases} \tag{3.1.6}$$

where $\rho_k = \frac{g_k^\top d_{k-1}}{\|g_k\|^2}$. Therefore, for $k \geq 1$

$$\begin{aligned} d_k &= -g_k + \beta_k^{CR}(d_{k-1} - \rho_k g_k), \\ &= -g_k + (\beta_k^{RMIL} + \theta_k(\beta_k^{hSM} - \beta_k^{RMIL}))(d_{k-1} - \rho_k g_k), \end{aligned}$$

that is,

$$d_k = -g_k + (\beta_k^{RMIL} + \theta_k \frac{2 g_k^\top g_{k-1}}{\|d_{k-1}\|^2})(d_{k-1} - \rho_k g_k). \tag{3.1.7}$$

Now, we need to consider the parameter θ_k. It is selected such that the search direction d_k satisfies also the following conjugacy condition

$$y_{k-1}^\top d_k = 0. \tag{3.1.8}$$

From (3.1.7) and (3.1.8), we get

$$0 = -y_{k-1}^\top g_k + (\beta_k^{RMIL} + \theta_k \frac{2 g_k^\top g_{k-1}}{\|d_{k-1}\|^2})(y_{k-1}^\top d_{k-1} - \rho_k y_{k-1}^\top g_k),$$

then

$$\begin{aligned} \theta_k &= \frac{y_{k-1}^\top g_k - \beta_k^{RMIL}(y_{k-1}^\top d_{k-1} - \rho_k y_{k-1}^\top g_k)}{2\frac{g_k^\top g_{k-1}}{\|d_{k-1}\|^2}(y_{k-1}^\top d_{k-1} - \rho_k y_{k-1}^\top g_k)}, \\ &= \frac{\zeta_k - \beta_k^{RMIL}\lambda_k}{\eta_k \lambda_k}, \end{aligned} \tag{3.1.9}$$

where $\zeta_k = y_{k-1}^\top g_k$, $\lambda_k = y_{k-1}^\top d_{k-1} - \rho_k y_{k-1}^\top g_k$ and $\eta_k = 2\frac{g_k^\top g_{k-1}}{\|d_{k-1}\|^2}$.

During the search process, we set $\theta_k = 1$ if $\theta_k > 1$; otherwise, we set $\theta_k = 0$ if for such an iteration we have $\theta_k < 0$ or $\eta_k \lambda_k = 0$. Algorithm 11 below summarizes the main steps of the proposed method.

Algorithm 11: The CR algorithm.

Step 0: (Initialization) Select $x_0 \in \mathbb{R}^n$ and the parameters $0 < \delta < \sigma < 1$, $\epsilon > 0$. Compute $h(x_0)$, $g_0 = \nabla h(x_0)$ and $d_0 = -g_0$. Set $k = 0$.

Step 1: If $\|g_k\| \leq \epsilon$, then stop; otherwise:
- Compute the step-length $\alpha_k > 0$ along the direction d_k using the strong Wolfe line search technique (3.0.4).
- Put $x_{k+1} = x_k + \alpha_k d_k$ and set $k = k + 1$.

Step 2: Compute the parameter θ_k: if $\eta_k \lambda_k = 0$ put $\theta_k = 0$, otherwise compute θ_k following the equation (3.1.9).

Step 3: β_k **computation:** β_k is computed following the equation (3.1.5) for $\theta_k \in (0,1)$; otherwise set $\beta_k = \beta_k^{RMIL}$ if $\theta_k \leq 0$ and, if $\theta_k \geq 1$, set $\beta_k = \beta_k^{hSM}$.

Step 4: **Search direction computation:** if the restart criterion of Powell $|g_k^\top g_{k-1}| \geq 0.2\|g_k\|^2$ holds, set $d_k = -g_k$; otherwise d_k is computed as in (3.1.6) and repeat Step 1.

3.1.1 The sufficient descent condition

It is well known that the sufficient descent property is crucial for the global convergence to hold. The next lemma deals with this issue, moreover it shows it is independent of any line search.

Lemma 3.1.1 *Let the sequences $\{g_k\}_{k\in\mathbb{N}}$ and $\{d_k\}_{k\in\mathbb{N}}$ be given by CR algorithm, then*

$$g_k^\top d_k = -\|g_k\|^2, \tag{3.1.10}$$

i.e. the direction d_k satisfies the sufficient descent condition.

Proof. It is clear that for $k=0$, the equation (3.1.10) is satisfied, that is $g_0^\top d_0 = -\|g_0\|^2$. Now for $k\geq 1$, we have

$$d_k = -g_k + \beta_k^{CR}(d_{k-1} - \rho_k g_k),$$

taking the inner product with $g_k^\top$ we get

$$\begin{aligned} g_k^\top d_k &= -\|g_k\|^2 - \beta_k^{CR} g_k^\top g_k \frac{g_k^\top d_{k-1}}{\|g_k\|^2} + \beta_k^{CR} g_k^\top d_{k-1}, \\ &= -\|g_k\|^2 - \beta_k^{CR} g_k^\top d_{k-1} + \beta_k^{CR} g_k^\top d_{k-1}, \\ &= -\|g_k\|^2, \end{aligned}$$

which finishes the proof. ■

3.1.2 The global convergence

In order to prove the global convergence of the CR method, we need the following Lemma.

Lemma 3.1.2 *Assume that Assumptions 1 and 2 hold. If the step-length α_k satisfies the strong Wolfe conditions (3.0.4) and d_k is a descent direction, then*

$$\alpha_k \geq \frac{(\sigma-1)}{L}\frac{d_k^\top g_k}{\|d_k\|^2}. \tag{3.1.11}$$

Proof. From the computation

$$\begin{aligned} (\sigma-1) d_k^\top g_k &\leq d_k^\top (g_{k+1} - g_k), \\ &\leq L\alpha_k \|d_k\|^2, \end{aligned}$$

it follows that,

$$\alpha_k \geq \frac{(\sigma-1)}{L}\frac{d_k^{\top} g_k}{\|d_k\|^2}.$$

■

Note that from (3.0.4), (3.1.10) and (3.1.11), it is clear that $\alpha_k \neq 0$. Hence, a constant $\gamma > 0$ must exist such that $\alpha \geq \gamma > 0$, for all $k \geq 0$.

Now, we are in the position to deal with the global convergence result

Theorem 3.1.1 *Suppose that Assumptions 1 and 2 hold. Let $\{g_k\}_{k\in\mathbb{N}}$ and $\{d_k\}_{k\in\mathbb{N}}$ be the sequences generated by CR algorithm; then*

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{3.1.12}$$

Proof. Assume that (3.1.12) is false, then a constant $C > 0$ exists such that

$$\|g_k\| \geq C,\ k \in \mathbb{N}. \tag{3.1.13}$$

Let $D = \max\{\|x-y\| : x, y \in \mathscr{N}\}$ be the diameter of the level set $\mathscr{S}$. By the Lipschitz continuity of g we have

$$\|g_k - g_{k-1}\| \leq L\|x_k - x_{k-1}\| \leq LD.$$

From (3.1.5), we have

$$\begin{aligned} |\beta_k^{CR}| =& |(1-\theta_k)\beta_k^{RMIL} + \theta_k \beta_k^{hSM}|, \\ \leq& |\beta_k^{RMIL}| + |\beta_k^{hSM}|. \end{aligned}$$

On the other hand, from [56] and [62] we have

$$|\beta_k^{RMIL}| \leq \frac{\|g_k\|\|y_{k-1}\|}{\|d_{k-1}\|^2} \leq \frac{rLD}{B^2} = G1,$$

$$|\beta_k^{hSM}| \leq \frac{\|g_k\|\|g_k + g_{k-1}\|}{\|d_{k-1}\|^2} \leq \frac{rA}{B^2} = G2,$$

so that

$$|\beta_k^{CR}| \leq G1 + G2 = G. \tag{3.1.14}$$

Thus, since $\forall k \in \mathbb{N}, \alpha \geq \gamma > 0$, then from (3.1.6), (3.1.14) and (1.3.4) it follows that,

$$\begin{aligned}\|d_k\| &\leq \|g_k\| + |\beta_k^{CR}|(\|d_{k-1}\| + \|\rho_k\|\|g_k\|) \\ &= \|g_k\| + 2G\|d_{k-1}\| \\ &\leq r + 2G\frac{\|x_k - x_{k-1}\|}{|\alpha_{k-1}|} \\ &\leq r + \frac{2GD}{\gamma} = E.\end{aligned}$$

Therefore,

$$\sum_{k\geq 0} \frac{1}{\|d_k\|^2} \geq \frac{1}{E^2}\sum_{k\geq 0} 1 = +\infty. \tag{3.1.15}$$

From equations (3.1.10) and (3.1.13), and Theorem 1.3.5, we have

$$C^4 \sum_{k\geq 0} \frac{1}{\|d_k\|^2} \leq \sum_{k\geq 0} \frac{\|g_k\|^4}{\|d_k\|^2} = \sum_{k\geq 0} \frac{(g_k^\top d_k)^2}{\|d_k\|^2} < +\infty,$$

which is a contradiction, so the assertion (3.1.12) is true. ■

3.2 Numerical Experiments

We present here a series of numerical results concerning the CR method applied on a collection of 34 functions with 100 test problems chosen from [8], as specified in Table 3.1, using dimensions ranging from 2 to 60000, Image Restoration problems are also considered. All numerical experiments are implemented in the scientific software MATLAB version R2015a, and run on PC with Intel(R) Core i3-4005U CPU 1.70 GHz and 4.00 RAM.

In the first part of this section we compare the performance of the proposed method with seven conjugate gradient methods that are

- The PRP method proposed by Polyak-Polak-Ribière [53].
- The EPF method proposed by Mtagulwa and Kaelo [50].
- The RMIL method proposed by Rivaie et al. [56].
- The hSM* method proposed by Sulaiman et al [62].
- The THCG+ method proposed by Lotfi and Hosseini [47].

Table 3.1: List of test problems.

Function	Dimension n	Function	Dimension n
Extended White and Holst	4000	Extended Maratos	500
Extended White and Holst	5000	Extended Maratos	700
Extended White and Holst	6000	Extended Maratos	1000
Extended Rosenbrock	1200	Extended Maratos	1500
Extended Rosenbrock	3000	POWER	2
Extended Rosenbrock	4000	Extended Quadratic Penalty QP1	50
Extended Rosenbrock	5000	Extended Quadratic Penalty QP1	100
Extended Freudenstein and Roth	9000	Extended Quadratic Penalty QP1	700
Extended Freudenstein and Roth	10000	Extended Quadratic Penalty QP1	1000
Extended Freudenstein and Roth	20000	Extended Quadratic Penalty QP1	1500
Extended Freudenstein and Roth	50000	Quadratic QF2	5000
Raydan 2	1000	Quadratic QF2	7000
Raydan 2	1500	Quadratic QF2	9000
Extended Tridiagonal 1	80	Extended Quadratic Penalty QP2	40
Extended Tridiagonal 1	90	Extended Quadratic Penalty QP2	60
Generalized Tridiagonal 1	10	Extended Quadratic Penalty QP2	70
Generalized Tridiagonal 1	20	ENGVAL1	1500
Generalized Tridiagonal 1	30	ENGVAL1	1600
Diagonal 3	6	ENGVAL1	1800
Diagonal 4	30000	Quartic	5000
Diagonal 4	40000	Quartic	8000
Diagonal 4	50000	Quartic	9000
Diagonal 4	60000	Quartic	10000
Diagonal 5	1000	HIMMELBH	2000
Diagonal 5	1500	HIMMELBH	2500
Diagonal 5	2000	HIMMELBH	2700
Diagonal 5	2500	HIMMELBH	3000
Diagonal 7	700	Extended BD1	2000
Diagonal 7	1500	Extended BD1	3000
Diagonal 7	2000	Extended BD1	5000
Diagonal 7	7000	Extended PSC1	2
Diagonal 8	1000	Extended PSC1	4
Diagonal 8	1500	Extended PSC1	6
Diagonal 8	2000	Extended DENSCHNF	1500
Diagonal 8	2500	Extended DENSCHNF	2000
Extended Himmelblau	9000	Extended DENSCHNF	2500
Extended Himmelblau	10000	Extended DENSCHNF	3000
FLETCHCR	2	Arwhead	50
FLETCHCR	4	Arwhead	70
NONSCOMP	2	Arwhead	150
Extended DENSCHNB	2000	Arwhead	200
Extended DENSCHNB	3000	HIMMELBG	50000
Extended DENSCHNB	5000	HIMMELBG	60000
Extended DENSCHNB	6000	LIARWHD	4000
Generalized Rosenbrok	2	LIARWHD	5000
Extended Hiebert	70	LIARWHD	5500
Extended Hiebert	500	LIARWHD	20000
Extended Hiebert	700	Hager	2
Extended Hiebert	1000	Hager	10
Almost Perturbed Quadratic	2	DIXON3DQ	2

- The ADHCG1 and ADHCG2 methods proposed by Livieris et al.[46].

The methods PRP, EPF, RMIL, hSM* and THCG+ are implemented using the strong Wolfe conditions, whereas ADHCG1 and ADHCG2 are implemented using the weak Wolfe conditions, by setting $\delta = 10^{-4}$ and $\sigma = 10^{-3}$. In this comparison, for each test function, the same initial point is chosen for these methods and every computation is terminated when a point x_k satisfying $\|g_k\|_\infty \leq 10^{-6}$ is found within 2000 iterations and whose calculation time does not exceed 500 seconds; otherwise, the computation is considered as a failure.

Throughout the numerical results, in Figures 3.1-3.4 we compare the performance of CR method with PRP, EPF, RMIL, hSM*, THCG+, ADHCG1 and ADHCG2 methods using the logarithmic performance profile of Dolan and Moré [23], relative to the number of iterations, function evaluations, gradient evaluations and CPU-time. For a solver s we define the ratio

$$r_{\mathrm{P},s} = \frac{N_{\mathrm{P},s}}{\min\{N_{\mathrm{P},s} : s \in S\}},$$

where $N_{\mathrm{P},s}$ denotes either the number of iterations, number of function (gradient) evaluations, or CPU-time required by the solver s to solve a problem P. If a solver s does not solve the problem P, the ratio $r_{\mathrm{P},s}$ is assigned a large number. The logarithmic performance profile for each solver s is defined as follows

$$\rho_s(\tau) = \frac{\text{number of problems where } \log_2(r_{\mathrm{P},s}) \leq \tau}{\text{total number of problems}},$$

For each method, we plot the fraction $\rho_s(\tau)$ of problems for which the method has a number of iterations (resp. number of function (gradient) evaluations and CPU-time) that is within a factor τ and the top curve in the plot corresponds to the method that solves most problems within a factor τ, for more details see [23].

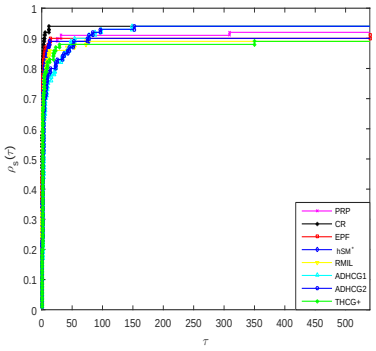

Figure 3.1: CPU Time performance profile.

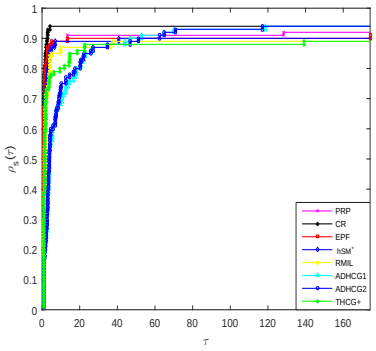

Figure 3.2: Iterations performance profile.

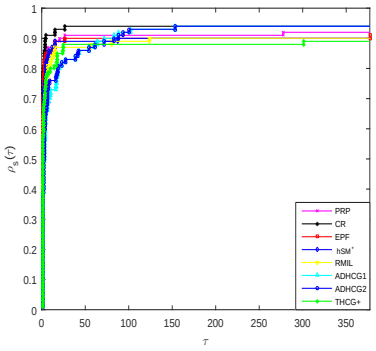

Figure 3.3: Function evaluations performance profile.

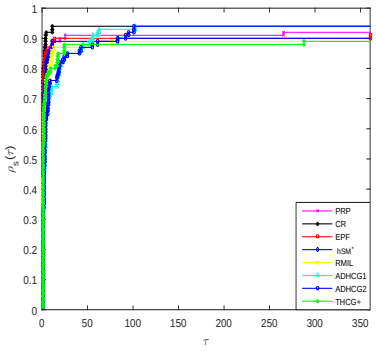

Figure 3.4: Gradient evaluations performance profile.

Figures 3.1-3.4 show that the curves of the methods CR, ADHCG1 and ADHCG2 dominate the other curves by solving 94% of the test problems successfully, with superiority to the CR method since it is faster then ADHCG1 and ADHCG2 on 78% of the test problems. The PRP and EPF methods have respectively the fourth and fifth best performances with 92% and 91% of test problems, followed by hSM* with 90% of the test problems, whereas RMIL and THCG+ score about 89%. These outcomes demonstrate that the CR method is competitive and converges quickly in the majority of the test problems.

3.2.1 Image restoration problems

Image restoration is of interest in optimization fields, it aims to recover the original image from an image damaged by impulse noises; its mathematical formulation can be found in [36]. In this subsection, we compare the performance of CR algorithm with the variants stud-

ied in the above subsection to solve image restoration problems. In this study, the images of Man.png (512×512), Lena.jpg (512×512), Boat.png (512×512) and Bridge.bmp (512×512) are selected as test images. The image quality is measured by the parameters: Iter (number of iterations), CPU-time, PSNR (Peak Signal-to-Noise Ratio) and Err (relative error) given by the following formulas:

$$PSNR = 10\log_{10}\frac{M\times N\times 255^2}{\sum_{i,j}(x^r_{i,j}-x^*_{i,j})^2}, \qquad Err = \frac{\|x^r-x^*\|}{\|x^*\|},$$

where $x^r_{i,j}$ and $x^*_{i,j}$ denote respectively the pixel values of the restored image and of the original one, M and N are the sizes of the image. The algorithm that has a large PSNR with small CPU-time and Err is chosen as the best one. The setting parameters are similarly chosen as in the above subsection, and each algorithm will stop as one of the following conditions is fulfilled

$$Iter > 300 \quad \text{or} \quad \frac{|h(x_{k+1}-h(x_k)|}{|h(x_k)|} < 10^{-4}.$$

The detailed performances for the Man, Lena, Boat and Bridge with 30 % and 70 % of salt-and-pepper noise are illustrated respectively in Figures 3.5 and 3.6. The obtained numerical results for the number of iterations, CPU-time, PSNR and the corresponding relative error are displayed in Tables 3.3 and 3.2 where the best results are styled in bold.

Inspect on of Figures 3.5 and 3.6 and the results obtained from Tables 3.3 and 3.2 shows a satisfactory performance of the CR algorithm. Indeed, it can be seen from the bold values in Table 3.3 that the proposed algorithm succeeds in restoring the majority of test images with higher PSNR values and overall needs less CPU time.

Figure 3.5: The noisy images with 30% salt-and-pepper (first row) and the restored images by EPF (second row), RMIL (third row), PRP (forth row), hSM* (fifth row), CR (sixth row), THCGP (seventh row), ADHCG1 (eighth row) and ADHCG2 (last row).

Figure 3.6: The noisy images with 70% salt-and-pepper (first row) and the restored images by EPF (second row), RMIL (third row), PRP (forth row), hSM* (fifth row), CR (sixth row), THCGP (seventh row), ADHCG1 (eighth row) and ADHCG2 (last row).

Methods/Images		Man	Lena	Boat	Bridge
EPF	Iter	18	19	18	29
	CPU	17.7664	17.9430	17.6929	15.5511
	PSNR	31.5369	37.7279	33.6354	28.5928
	Err	0.0551	0.0286	0.0385	0.0751
RMIL	Iter	15	**15**	**9**	17
	CPU	21.8868	21.2731	16.6706	21.1122
	PSNR	31.4990	37.5832	32.6893	28.5734
	Err	0.055368	0.029076	0.042920	0.075255
PRP	Iter	**11**	**15**	11	**14**
	CPU	19.5940	17.6209	19.3073	16.3226
	PSNR	31.3375	**37.7638**	33.1674	28.5131
	Err	0.056408	**0.028477**	0.040622	0.075779
hSM*	Iter	15	17	16	15
	CPU	17.6889	18.1050	17.5557	15.3714
	PSNR	31.5501	37.7225	33.6823	28.5813
	Err	0.055043	0.028613	0.038284	0.075186
CR	Iter	17	**15**	17	17
	CPU	**13.8810**	14.2103	13.5648	**13.5886**
	PSNR	**31.5597**	37.7533	33.6639	28.5931
	Err	**0.054983**	0.028512	0.038365	0.075084
ADHCG1	Iter	46	43	44	44
	CPU	16.6226	16.9791	17.3495	16.4667
	PSNR	31.5332	37.5717	33.6280	**28.6268**
	Err	0.055151	0.029114	0.038524	**0.074794**
ADHCG2	Iter	46	43	44	44
	CPU	16.5253	16.9898	17.5813	16.4783
	PSNR	31.5332	37.5717	33.6280	**28.6268**
	Err	0.055151	0.029114	0.038524	**0.074794**
THCG+	Iter	29	21	22	25
	CPU	14.5561	**12.7482**	**13.1498**	13.7842
	PSNR	31.5542	37.7427	**33.6844**	28.4667
	Err	0.055017	0.028547	**0.038275**	0.076185

Table 3.2: Numerical results for image restoration problems with 30% salt-and-pepper.

Methods/Images		Man	Lena	Boat	Bridge
EPF	Iter	24	30	28	31
	CPU	27.6617	38.0655	34.3126	36.7456
	PSNR	26.2376	31.7051	28.2103	24.5522
	Err	0.1015	0.0572	0.0719	0.1196
RMIL	Iter	23	22	26	**15**
	CPU	51.9098	55.7232	62.9898	51.2787
	PSNR	26.1395	31.1952	28.2408	22.4305
	Err	0.102621	0.060664	0.071630	0.152643
PRP	Iter	**20**	**5**	**15**	**15**
	CPU	49.1265	28.6989	37.0087	54.1025
	PSNR	26.1119	15.5820	27.5034	22.0518
	Err	0.102948	0.366091	0.077976	0.159447
hSM*	Iter	23	23	18	19
	CPU	34.1524	34.3400	32.1868	31.0612
	PSNR	26.2949	31.6644	28.2232	24.4001
	Err	0.100802	0.057473	0.071775	0.121674
CR	Iter	**20**	30	23	16
	CPU	31.6035	32.9323	36.8002	**15.9113**
	PSNR	26.2393	**31.7190**	28.2483	**28.5640**
	Err	0.101449	**0.057114**	0.071567	**0.075337**
ADHCG1	Iter	70	59	65	67
	CPU	40.8277	35.0237	37.0822	36.0258
	PSNR	26.1809	31.4986	28.1806	24.3848
	Err	0.102132	0.058581	0.072128	0.121889
ADHCG2	Iter	70	59	65	67
	CPU	40.0976	34.8940	37.0464	36.1044
	PSNR	26.1809	31.4986	28.1806	24.3848
	Err	0.102132	0.058581	0.072128	0.121889
THCG+	Iter	29	27	31	26
	CPU	**26.1769**	**25.4340**	**26.0725**	24.0349
	PSNR	**26.3218**	31.6147	**28.3121**	24.4231
	Err	**0.100490**	0.057803	**0.071044**	0.121352

Table 3.3: Numerical results for image restoration problems with 70% salt-and-pepper.

CHAPTER

4 An Efficient Hybrid Conjugate Gradient Method for Unconstrained Optimization and Image Restoration Problems

Conjugate gradient (CG) methods stand out as popular and efficient techniques extensively employed for solving unconstrained optimization problems, particularly for large-scale cases, due to their convergence properties and low computation cost. In this study, we address the following nonlinear unconstrained problem

$$\min_{x\in\mathbb{R}^n} h(x), \tag{4.0.1}$$

where the function $h:\mathbb{R}^n \longrightarrow \mathbb{R}$ is differentiable continuous with gradient $g(x)=\nabla h(x)$. The common denominator of all CG methods is to generate a sequence of points $\{x_k\}_{k\in\mathbb{N}} \subset \mathbb{R}^n$ starting from an initial point $x_0 \in \mathbb{R}^n$ following the scheme

$$x_{k+1} = x_k + \alpha_k d_k, \tag{4.0.2}$$

where d_k is a descent direction for h at x_k and $\alpha_k > 0$ is a step-length which is determined through a one-dimensional search procedure known as the 'line search' where

$$\alpha_k = \text{argmin}_{\alpha\geq 0} h(x_k+\alpha d_k).$$

Here, the step-length is computed using strong Wolfe line search, i.e α_k satisfies

$$\begin{aligned} h(x_k+\alpha_k d_k)-h(x_k) &\leq \delta\alpha_k g_k^\top d_k, \\ |\nabla h(x_k+\alpha_k d_k)^\top d_k| &\leq -\sigma g_k^\top d_k, \end{aligned} \tag{4.0.3}$$

where $0<\delta<\frac{1}{2}$, $\delta<\sigma<1$. The search direction d_k is usually defined by the following formula

$$\begin{aligned} d_0 &= -g_0, \\ d_{k+1} &= -g_{k+1}+\beta_k d_k \quad k\geq 0, \end{aligned} \tag{4.0.4}$$

where the parameter β_k is a scalar which determines the different conjugate gradient methods.

In this chapter, a novel conjugate gradient method is introduced to solve nonlinear unconstrained optimization problems. Based on the combination of PRP (Polak-Ribière-Polyak), HRM (Hamoda-Rivaie-Mamat), and NMFR (New modified Fletcher-Reeves) methods, this method produces a descent direction without depending on any line search. Moreover, it enjoys global convergence under mild assumptions and is applied on various standard test problems as well as image processing. The numerical results indicate that the proposed method outperforms several existing methods in terms of efficiency. Furthermore, the proposed approach has been successfully applied for image restoration.

4.1 The Proposed Algorithm

In this section, Our aim is to elaborate on an efficient hybrid conjugate gradient method, based on the combination of PRP (Polak-Ribière-Polyak)[53], HRM (Hamoda-Rivaie-Mamat)[33] and NMFR (New modified Fletcher-Reeves)[1]. The new hybrid choice for the parameter β_k is as follows

$$\beta_k^{cPHN}=\begin{cases}\beta_k^{PRP} & \text{if} \quad 0\leq\beta_k^{PRP}\leq\beta_k^{FR},\\ (1-\phi_k)\beta_k^{HRM}+\phi_k\beta_k^{NMFR} & \text{otherwise,}\end{cases} \tag{4.1.5}$$

where

$$\beta_k^{PRP}=\frac{g_{k+1}^{\top}y_k}{\|g_k\|^2},$$

$$\beta_k^{HRM}=\frac{g_{k+1}^{\top}\left(g_{k+1}-\frac{\|g_{k+1}\|}{\|g_k\|}g_k\right)}{\mu_1\|g_k\|^2+(1-\mu_1)\|d_k\|^2},$$

and

$$\beta_k^{NMFR}=\frac{\|g_{k+1}\|^2}{\mu_2\|g_k\|^2+(1-\mu_2)\|d_k\|^2},$$

here, $0 \leq \mu_1, \mu_2, \phi_k \leq 1$. The search direction d_k is computed as follows

$$\begin{cases} d_0 & = -g_0, \\ d_{k+1} & = -g_{k+1} + \beta_k^{cPHN}(d_k - \rho_k g_{k+1}) \quad k \geq 0, \end{cases} \tag{4.1.6}$$

where $\rho_k = \frac{d_k^\top g_{k+1}}{\|g_{k+1}\|^2}$. The parameter ρ_k allows to generate a search direction that satisfies the sufficient descent condition independently of any line search and makes the next search direction to approach the steepest direction, which is crucial to achieve the global convergence.

Our incentive for choosing the parameter ϕ_k is that the search direction d_{k+1} should fulfill the famous D-L conjugacy condition [21]

$$d_{k+1}^\top y_k = -t\, s_k^\top g_{k+1}, \quad t > 0, \tag{4.1.7}$$

where $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$.

If $\beta_k^{cPHN} = (1-\phi_k)\beta_k^{HRM} + \phi_k \beta_k^{NMFR}$, it follows from (4.1.6) that the search direction d_{k+1} can be written as follows

$$d_{k+1} = -g_{k+1} + \big(\beta_k^{HRM} + \phi_k(\beta_k^{NMFR} - \beta_k^{HRM})\big)(d_k - \rho_k g_{k+1});$$

then by equation (4.1.7) we get

$$-t\, s_k^\top g_{k+1} = -g_{k+1}^\top y_k + \big(\beta_k^{HRM} + \phi_k(\beta_k^{NMFR} - \beta_k^{HRM})\big)\big(d_k^\top y_k - \rho_k g_{k+1}^\top y_k\big),$$

hence,

$$\phi_k = \frac{\tau_k - \beta_k^{HRM}\zeta_k}{\lambda_k \zeta_k}, \tag{4.1.8}$$

where $\tau_k = -t\, s_k^\top g_{k+1} + g_{k+1}^\top y_k$, $\zeta_k = d_k^\top y_k - \rho_k g_{k+1}^\top y_k$ and $\lambda_k = \beta_k^{NMFR} - \beta_k^{HRM}$.

During the search process, if for such an iteration we have $\lambda_k \zeta_k = 0$ or $\phi_k < 0$ we set $\phi_k = 0$ and in the case where $\phi_k > 1$ we set $\phi_k = 1$. The main steps of the proposed method are outlined in the algorithm 12 below.

4.1.1 The sufficient descent condition

The next lemma shows that the search direction produced by cPHN method is independent of any line search.

Algorithm 12: The cPHN algorithm

Step 0: (Initialization) Select $x_0 \in \mathbb{R}^n$ and the parameters $0 < \delta < \sigma < 1$, $\epsilon > 0$. Compute $h(x_0)$, $g_0 = \nabla h(x_0)$ and $d_0 = -g_0$. Set $k = 0$.

Step 1: If $\|g_k\| \leq \epsilon$, then stop; otherwise:
- Compute the step-length α_k using the strong Wolfe technique (4.0.3).
- Put $x_{k+1} = x_k + \alpha_k d_k$.

Step 2: Compute the parameter ϕ_k: if $\lambda_k \zeta_k = 0$ put $\phi_k = 0$, otherwise compute ϕ_k following the equation (4.1.8).

Step 3: β_k **computation:** β_k is computed following the equation(4.1.5).

Step 4: **Search direction computation:** if the restart criterion of Powell $|g_{k+1}^{\top} g_k| \geq 0.2 \|g_{k+1}\|^2$ holds, then set $d_{k+1} = -g_{k+1}$; otherwise d_{k+1} is computed as in (4.1.6) and repeat Step 1.

Lemma 4.1.1 *Suppose that the cPHN algorithm generates the sequences* $\{g_k\}_{k\in\mathbb{N}}$ *and* $\{d_k\}_{k\in\mathbb{N}}$, *then the direction* d_k *satisfies the sufficient descent condition, i.e.*

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 \tag{4.1.9}$$

Proof. It is clear that the relation (4.1.9) holds when $d_0 = -g_0$. Now for $k \geq 0$, we have

$$d_{k+1} = -g_{k+1} + \beta_k^{cPHN}(d_k - \rho_k g_{k+1}),$$

by multiplying both sides of the equation by $g_{k+1}^{\top}$, we get

$$\begin{aligned} g_{k+1}^{\top} d_{k+1} &= -\|g_{k+1}\|^2 - \beta_k^{cPHN} g_{k+1}^{\top} d_k + \beta_k^{cPHN} g_{k+1}^{\top} d_k \\ &= -\|g_{k+1}\|^2, \end{aligned}$$

which completes the proof. ■

4.1.2 The convergence analysis

Lemma 4.1.2 *Assume that Assumptions 1 and 2 hold. If the step-length α_k satisfies the strong Wolfe conditions (4.0.3) and d_k is a descent direction, then*

$$\alpha_k \geq \frac{(\sigma-1)d_k^\top g_k}{L\|d_k\|^2} \quad \forall k \in \mathbb{N}. \tag{4.1.10}$$

Proof. Refer to the proof of Lemma 3.1.2 ■

According to the strong Wolfe conditions (4.0.3) and Lemma 4.1.2, it follows that our α_k is strictly positive for all $k \in \mathbb{N}$, hence $\exists \gamma > 0$ such that $\alpha_k \geq \gamma > 0$ for all $k \in \mathbb{N}$.

To prove the global convergence of the proposed method, we use the following lemma due to Dai et al. [20]. It applies to any conjugate gradient method that relies on the strong Wolfe conditions.

Lemma 4.1.3 *Let Assumptions 1 and 2 hold. Consider an iterative method in the form (4.0.2) and (4.0.4), where d_k is a descent direction and α_k satisfies the strong Wolfe conditions (4.0.3). If*

$$\sum_{k\geq 1} \frac{1}{\|d_k\|^2} = +\infty,$$

then

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

Now, we are in the position to state our result.

Theorem 4.1.1 *Let $\{g_k\}_{k\in\mathbb{N}}$ and $\{d_k\}_{k\in\mathbb{N}}$ be the sequences produced by the cPHN method, where the step-length α_k is computed using the strong Wolfe conditions. Suppose that Assumptions 1 and 2 hold, then*

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{4.1.11}$$

Proof. Suppose that the assertion (4.1.11) is false, then there exists a constant $C > 0$ such that

$$\|g_k\| \geq C, \quad \forall k \in \mathbb{N}. \tag{4.1.12}$$

Let $D = \max\{\|x-y\| : x, y \in \mathcal{N}\}$ be the diameter of the level set $\mathcal{S}$. From [33] and [1] we have

$$0 \leq \beta_k^{HRM} \leq \frac{2\|g_{k+1}\|^2}{\mu_1 \|g_k\|^2}$$

and

$$0 \leq \beta_k^{NMFR} \leq \frac{\|g_{k+1}\|^2}{\mu_2 \|g_k\|^2},$$

where $0 \leq \mu_1, \mu_2 \leq 1$, it follows from equation (4.1.5) that

$$0 \leq \beta_k^{cPHN} \leq \beta_k^{PRP} + (1-\phi_k)\beta_k^{HRM} + \phi_k \beta_k^{NMFR} \leq \beta_k^{FR} + \beta_k^{HRM} + \beta_k^{NMFR},$$

then

$$\beta_k^{cPHN} \leq \frac{\|g_{k+1}\|^2}{\|g_k\|^2} + \frac{2\|g_{k+1}\|^2}{\mu_1 \|g_k\|^2} + \frac{\|g_{k+1}\|^2}{\mu_2 \|g_k\|^2} = \left(1 + \frac{2}{\mu_1} + \frac{1}{\mu_2}\right) \frac{\|g_{k+1}\|^2}{\|g_k\|^2}.$$

By setting $\xi = 1 + \frac{2}{\mu_1} + \frac{1}{\mu_2}$, we get

$$0 \leq \beta_k^{cPHN} \leq \xi \frac{\|g_{k+1}\|^2}{\|g_k\|^2} \leq \xi \frac{r^2}{C^2}. \tag{4.1.13}$$

Therefore, from (4.1.6) and (1.3.4) it follows that

$$\begin{aligned}
\|d_{k+1}\| &\leq \|g_{k+1}\| + |\beta_k^{cPHN}|(\|d_k\| + \|\rho_k\|\|g_{k+1}\|) \\
&\leq \|g_{k+1}\| + 2\xi \frac{r^2}{C^2}\|d_k\| \\
&\leq r + 2\xi \frac{r^2}{C^2} \frac{\|x_{k+1} - x_k\|}{\alpha_k} \\
&\leq r + 2\xi \frac{r^2}{C^2} \frac{D}{\gamma}.
\end{aligned}$$

Hence,

$$\sum_{k \geq 0} \frac{1}{\|d_{k+1}\|^2} = +\infty,$$

according to Lemma 4.1.3 $\lim_{k \to \infty} \inf \|g_k\| = 0$, which contradicts the claim (4.1.12), so the assertion (4.1.11) is true. ■

4.2 Numerical Experiments

In this section, we present a series of computational performances concerning the cPHN method applied on 130 problems taken from [8], as outlined in Table 4.1, using an increasing number of dimensions $n = 2, 4, .., 80000$; Image processing problems are also presented. All codes are written and implemented in Matlab version R2015a, and run on PC with Intel(R) Core i3-4005U CPU 1.70 GHz and 4.00 RAM.

Table 4.1: List of test problems.

Function	Dimension n
Extended White and Holst	1000, 3000, 4000, 6000
Extended Rosenbrock	10, 20, 30, 100
Extended Freudenstein and Roth	1000, 4000, 9000, 50000, 80000
Extended DENSCHNF	10, 100, 10000, 50000, 70000
Extended Tridiagonal 1	300, 500, 700, 1000
Extended Himmelblau	4, 6, 8, 10
Extended DENSCHNB	5000, 6000, 7000, 9000
Extended quadratic exponential EP1	40000, 50000, 60000, 70000
Extended BD1	20000, 40000, 60000, 70000, 80000
Extended quadratic penalty QP1	4, 6, 8, 10
Extended quadratic penalty QP2	200
Extended PSC1	2, 6, 8, 100
Extended Maratos	9000, 9500, 10000, 15000
Generalized Rosenbrock	2
FLETCHCR	2
NONSCOMP	4
Almost Perturbed Quadratic	2,4, 6
Diagonal 1	2, 4, 10
Diagonal 2	2, 4, 10
Diagonal 3	2, 4
Diagonal 4	20000, 40000, 60000, 70000
Diagonal 5	2000, 2200, 2500
Diagonal 7	500, 700, 1000, 1500, 2000
Diagonal 8	100, 200, 300, 400, 500
Raydan 1	2
Raydan 2	1000, 4000, 50000,80000
Arwhead	70, 80, 100, 150
ENGVAL1	600, 700, 800
HIMMELLH	10, 50, 300, 500
HIMMELBG	3000, 6000, 30000, 50000, 80000
Generalized Tridiagonal 1	2
Perturbed quadratic diagonal	2
Perturbed Quadratic	2
POWER	2
QUARTC	2
DIXON3DQ	2, 4
LIARWHD	10, 20, 40, 50
Hager	50, 80, 150, 300
Quadratic QF1	9000, 20000, 50000, 70000, 80000
Quadratic QF2	100, 200

We begin by comparing the performance of the proposed method against

- The PRP$^+$ method introduced by Gilbert et al. [29],
- The HRM method with $\mu_1 = 0.4$ introduced by Hamoda et al. [33].
- The NPRP and NHS methods introduced by Zhang [64],
- The FR method introduced by Fletcher and Reeves. [27].
- The NMFR method with $\mu_2 = 0.4$ introduced by Abdelrahman et al [1].
- The CR method introduced by Souli et al. [60]

In this study, for each test problem, the same starting point is chosen for these methods, and every computation is stopped when a point x_k satisfying $\|g_k\|_\infty \leq 10^{-6}$ is found within 2000 iterations and whose CPU time is less than 500 seconds; otherwise, the computation is assigned as a failure. The step-lengths of all tested algorithms are determined using the strong Wolfe line search technique(4.0.3) with $\sigma = 10^{-3}$ and $\delta = 10^{-4}$.

Throughout the numerical results, in Figures 4.1-4.4 the global performances of the four methods are compared with cPHN (using their respective performance profiles relative to the number of iterations, function evaluations, gradient evaluations and CPU-time needed to reach the stopping criterion) under the logarithmic performance profile of Dolan and Moré. For a solver s we define the ratio

$$r_{\mathrm{P},s} = \frac{N_{\mathrm{P},s}}{\min\{N_{\mathrm{P},s} : s \in S\}}$$

where $N_{\mathrm{P},s}$ denotes either the number of iterations (resp. CPU-time or number of function (gradient) evaluations) requested by the solver s to solve a problem P. If a solver s does not solve the problem P, the ratio $r_{\mathrm{P},s}$ is assigned a large number r_M. The logarithmic performance profile for each solver s is defined as follows

$$\rho_s(\tau) = \frac{\text{number of problems where } \log_2(r_{\mathrm{P},s}) \leq \tau}{\text{total number of problems}}.$$

For each method, we plot the fraction $\rho_s(\tau)$ of problems for which the method has a number of iterations (resp. number of function (gradient) evaluations and CPU-time) that is within a

factor τ. The highest curve in the plot corresponds to the method that solves most problems within a factor τ, for more details see [23]

Figures 4.1-4.4 illustrate the fact that the cPHN outperforms the others, notably it is fastest for about 50% of the test problems and successfully solves about 97% of them, followed by PRP* with 95%. The methods FR and NMFR have respectively the third and the fourth best performance by solving about 95% and 94% of the test problems, NMFR has the fifth best performance by solving about 93%, whereas NPRP and NHS score respectively about 92% and 73%. These results demonstrate the competitiveness and rapid convergence of the cPHN method in the majority of testing problems.

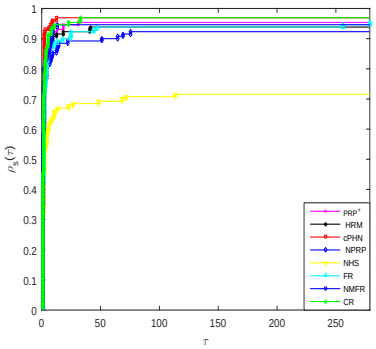

Figure 4.1: CPU Time performance profile.

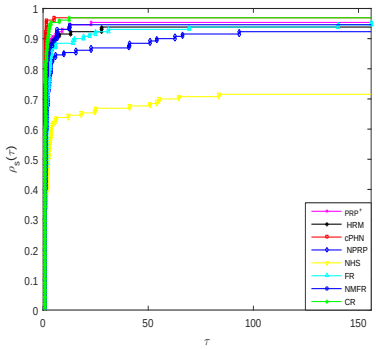

Figure 4.2: Iterations performance profile.

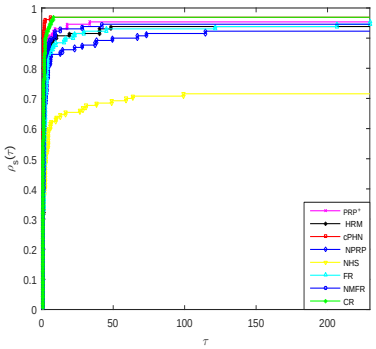

Figure 4.3: Function evaluations performance profile.

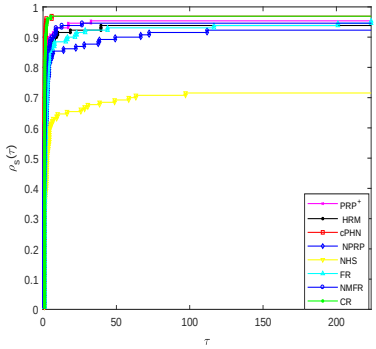

Figure 4.4: Gradient evaluations performance profile.

4.2.1 Image restoration problems

Image restoration problems are considered among the most difficult problems in optimization fields, they aim to restore the original image from an image corrupted by impulse noise. The mathematical formulation can be found in [36]. In this study, the images of Man.png (512×512), Boat.png (512×512), Lena.jpg (512×512) and Bridge.bmp (512×512) are selected as test images for evaluating the effectiveness of the cPHN algorithm against the same variants used in the previous test comparison. The image quality is assessed by several parameters: CPU-time, Iter (number of iterations), PSNR (Peak Signal-to-Noise Ratio) and Err (relative error) using the following formulas:

$$PSNR = 10\log_{10}\frac{M\times N\times 255^2}{\sum_{i,j}(x^r_{i,j}-x^*_{i,j})^2}, \qquad Err = \frac{\|x^r-x^*\|}{\|x^*\|}$$

where M and N are the sizes of the image, $x^r_{i,j}$ represents the pixel values of the restored image and $x^*_{i,j}$ denotes the pixel values of the original one. The parameters of the proponent algorithms are set similarly to the previous test, and each computation will stop if one of the following criteria holds

$$Iter > 300 \quad \text{or} \quad \frac{|h(x_{k+1}-h(x_k)|}{|h(x_k)|} < 10^{-4}.$$

The performances for the Man, Boat, Lena and Bridge with 30 %, 50% and 70 % of salt-and-pepper noise are depicted respectively in Figures 4.5, 4.6 and 4.7. The numerical outcomes encompassing the number of iterations, CPU time, PSNR and corresponding relative error (err) are presented in Tables 4.2, 4.3, and 4.4, where the best results are highlighted in bold font. The algorithm that has a large PSNR with small CPU-time and Err is chosen as the best one.

Upon examining Figures 4.5 - 4.7 we see that the quality of the images restored by cPHN, FR, PRP$^+$, NMFR, HRM and CR is very similar while the images restored by NHS and NPRP is not good restored . The detailed numerical results are reported in Tables 4.2 - 4.4.

From the numerical results it becomes evident that the cPHN algorithm delivers good performance. Notably, the bold values in Tables 4.2 - 4.4 highlight the efficiency of the proposed algorithm as it achieves higher PSNR values and requires comparatively less CPU time for restoring the majority of the test images

Figure 4.5: The noisy images with 30% salt-and-pepper (first row) and the restored images by FR (second row), HRM (third row), NHS (fourth row), NMFR (fifth row), NPRP (sixth row),PRP^+ (seventh row), CR (eighth row) and cPHN (last row).

Figure 4.6: The noisy images with 50% salt-and-pepper (first row) and the restored images by FR (second row), HRM (third row), NHS (fourth row), NMFR (fifth row), NPRP (sixth row),PRP^+ (seventh row), CR (eighth row) and cPHN (last row).

Figure 4.7: The noisy images with 70% salt-and-pepper (first row) and the restored images by FR (second row), HRM (third row), NHS (fourth row), NMFR (fifth row), NPRP (sixth row),PRP^+ (seventh row), CR (eighth row) and cPHN (last row).

Methods/Images		Man	Boat	Lena	Bridge
FR	Iter	152	112	118	180
	CPU	59.2035	48.1470	46.7314	64.7920
	PSNR	28.9909	30.8528	30.7913	27.7053
	Err	0.073904	0.053026	0.063551	0.083165
NHS	Iter	26	26	25	23
	CPU	15.7869	14.9272	14.5193	14.4291
	PSNR	15.7231	17.5183	17.3303	16.2491
	Err	0.3405	0.2462	0.2993	0.3110
cPHN	Iter	32	25	32	35
	CPU	15.8053	15.2852	16.0294	16.5909
	PSNR	**31.5613**	**33.7160**	37.7570	28.4505
	Err	**0.054973**	**0.038136**	0.028499	0.076327
PRP^+	Iter	9	14	11	13
	CPU	**11.7255**	17.2765	12.4417	12.5381
	PSNR	29.8526	32.7850	36.7924	28.3708
	Err	0.066923	0.042450	0.031847	0.077031
NMFR	Iter	18	16	15	35
	CPU	13.5430	12.8837	13.1202	16.5120
	PSNR	31.5222	33.6216	37.6712	28.4505
	Err	0.055220	0.038552	0.028783	0.076327
HRM	Iter	13	14	16	19
	CPU	14.4333	16.1088	15.9510	18.3902
	PSNR	31.4316	33.5530	**37.7937**	28.5705
	Err	0.055800	0.038858	**0.028379**	0.075280
NPRP	Iter	**6**	**6**	**6**	**6**
	CPU	12.4714	**12.5510**	**11.1241**	**12.1049**
	PSNR	15.7089	17.5182	17.3155	16.2439
	Err	0.341010	0.246162	0.299857	0.311179
CR	Iter	17	**17**	15	17
	CPU	13.8810	13.5648	14.2103	13.5886
	PSNR	31.5597	33.6639	37.7533	**28.5931**
	Err	0.054983	0.038365	0.028512	**0.075084**

Table 4.2: Numerical results for image restoration problems with 30% salt-and-pepper.

Methods/Images		Man	Boat	Lena	Bridge
FR	Iter	114	166	119	36
	CPU	74.2768	95.7872	77.3813	29.8402
	PSNR	25.5428	27.9388	27.8919	21.9921
	Err	0.109918	0.074164	0.088735	0.160546
NHS	Iter	31	34	33	30
	CPU	24.3502	24.8263	24.8593	23.0874
	PSNR	13.4363	15.2070	14.7535	14.1568
	Err	0.4430	0.3212	0.4027	0.3957
cPHN	Iter	35	33	32	32
	CPU	23.0232	22.1550	22.8474	22.4948
	PSNR	**29.1641**	**31.1930**	**35.1330**	26.6833
	Err	**0.072445**	**0.050990**	**0.038551**	0.093549
PRP^+	Iter	**5**	17	17	16
	CPU	16.5329	22.4427	31.0590	21.0229
	PSNR	15.3421	30.9165	34.0432	26.5201
	Err	0.355716	0.052639	0.043705	0.095324
NMFR	Iter	22	17	17	18
	CPU	20.1598	**18.0440**	18.1020	17.7847
	PSNR	29.1302	31.0974	35.0066	26.5072
	Err	0.072728	0.051554	0.039116	0.095465
HRM	Iter	19	14	26	14
	CPU	31.6329	22.9351	33.0992	21.4267
	PSNR	29.0482	30.0439	35.0633	26.2812
	Err	0.073418	0.058202	0.038862	0.097982
NPRP	Iter	8	**10**	**8**	**8**
	CPU	**16.0684**	18.4926	**16.6391**	**16.2163**
	PSNR	13.4675	15.1712	14.7757	14.1672
	Err	0.441405	0.322532	0.401702	0.395223
CR	Iter	18	19	20	19
	CPU	25.0892	22.2550	20.8171	17.6200
	PSNR	29.1528	31.1018	35.0045	**26.7484**
	Err	0.072539	0.051527	0.039126	**0.092850**

Table 4.3: Numerical results for image restoration problems with 50% salt-and-pepper.

Methods/Images		Man	Boat	Lena	Bridge
FR	Iter	128	124	151	215
	CPU	85.8010	86.0845	108.1340	133.9843
	PSNR	23.9667	25.0664	27.6667	24.0103
	Err	0.131788	0.103230	0.091065	0.127260
NHS	Iter	34	37	34	31
	CPU	29.7937	30.6846	27.8669	22.3260
	PSNR	11.3935	12.9901	12.5870	12.1899
	Err	0.5604	0.4146	0.5168	0.4963
cPHN	Iter	33	32	27	34
	CPU	27.3060	25.3865	24.5562	26.9649
	PSNR	**26.3186**	**28.2960**	**31.7486**	24.5198
	Err	**0.100527**	**0.071175**	**0.056919**	0.120009
PRP^+	Iter	29	24	24	18
	CPU	50.3015	35.5985	42.2219	35.9517
	PSNR	26.1898	28.2480	31.5568	24.1335
	Err	0.102028	0.071570	0.058190	0.125468
NMFR	Iter	24	23	20	25
	CPU	24.9760	23.0873	**13.6592**	25.6359
	PSNR	26.2295	28.1877	28.5882	24.4006
	Err	0.101563	0.072068	0.075126	0.121667
HRM	Iter	10	26	20	25
	CPU	28.8301	49.0126	40.3861	48.7749
	PSNR	22.3136	28.1401	30.0993	24.2470
	Err	0.159416	0.072465	0.068821	0.123838
NPRP	Iter	**8**	**6**	**7**	**6**
	CPU	**22.5660**	**20.0892**	20.4119	19.4860
	PSNR	11.3863	13.0485	12.5869	12.1776
	Err	0.560914	0.411816	0.516821	0.496963
CR	Iter	20	23	30	16
	CPU	31.6035	36.8002	32.9323	**15.9113**
	PSNR	26.2393	28.2483	31.7190	**28.5640**
	Err	0.101449	0.071567	0.057114	**0.075337**

Table 4.4: Numerical results for image restoration problems with 70% salt-and-pepper.

Conclusion

In this work, we explored various innovative methodologies for tackling non linear optimization problems across different contexts. Each proposed approach contributes to the field by enhancing computational efficiency, theoretical robustness, and practical applicability.

In the first work, a logarithmic barrier interior point method incorporating a vector penalty term was developed. This method leverages minorant function which is related to a secant method to improve step-length computation, enhancing simplicity and efficiency. Numerical experiments confirmed its ability to outperform conventional line search methods in solving nonlinear convex problems.

In the second and final publications, novel CR and cPHN conjugate gradient methods were proposed. These methods generate descent directions independently of line search and ensure global convergence under mild assumptions. Extensive numerical results and applications to image restoration demonstrated their effectiveness and superiority over existing methods.

Bibliography

[1] A. Abdelrahman, M. Mohammed, O. O. Yousif, M. K. Elbashir, . Nonlinear Conjugate Gradient Coefficients with Exact and Strong Wolfe Line Searches Techniques. Journal of Mathematics, 2022.

[2] M. Achache. Multidimensional primal-dual path-following interior point methods for linear programming and linear complementarity problems. These de Doctorat d'Etat, Université Ferhat Abbas de Sétif, 19000 (2005).

[3] M. Achache. A new primal-dual path-following method for convex quadratic programming. Comput. Appl. Math. 25 (2006) 97–110.

[4] M. Achache. A weighted full-Newton step primal-dual interior point algorithm for convex quadratic optimization. Statistics, Optimization & Information Computing, 2(1) (2014) 21-32.

[5] M. Achache. N. Tabchouche. A full Nesterov-Todd step primal-dual path-following interior point algorithm for semidefinite linear complementarity problems. Croatian Operational Research Review,(2018) 37-50.

[6] M. Achache. A polynomial-time weighted path-following interior-point algorithm for linear optimization. Asian-European Journal of Mathematics, 13(02) (2020), 2050038.

[7] M. Al-Baali, Descent property and global convergence of the Fletcher-Reeves method with inexact line search, IMA J. Numer. Anal. 5 (1985) 121–124.

[8] N. Andrei, An unconstrained optimization test functions, Adv. Modeling Optim., 10(2008) 147–161.

[9] A. Antoniou, Wu-Sheng lu, "PRACTICAL OPTIMIZATION Algorithms and Engineering Applications", Department of Electrical and Computer Engineering, University of Victoria, Canada.

[10] M. S. Bazraa, H. D. Sherali and C. M. Shetty. Nonlinear Programming. Published by John Wiley & Sons, Inc., Hoboken, New Jersey, 2006.

[11] Y. Bendaas, M. Achache. A numerical study of an infeasible interior-point algorithm for convex quadratic semi-definite optimization. Journal of Numerical Analysis and Approximation Theory, 53(2)(2024) 199-217.

[12] M. Bierlaire, "Introduction à l'optimisation différentiable", press polytechniques et universitaires romandes, (2006).

[13] J. F. Bonnans, J. C. Gilbert, C. Lemarechal and C. Sagastizabal. Numerical Optimization: Theoretical and Practical Aspects. Mathematics and Applications, Vol. 27. Springer-Verlag, Berlin, 2003.

[14] S. Boyd and L. Vandenberghe, "Convex Optimization", Cambridge University Press, (2004).

[15] A. Cauchy et al. « Méthode générale pour la résolution des systemes d'équations simultanées ». In : Comp. Rend. Sci. Paris 25.1847 (1847), p. 536-538.

[16] S. Chaghoub, D. Benterki. A Logarithmic Barrier Method Based On a New Majorant Function for Convex Quadratic Programming. IAENG International Journal of Applied Mathematics, 51:3, IJAM_51_3_14, Volume 51, Issue 3 (2021).

[17] J.P. Crouzeix, A. Keraghel. W.S. Sandoval. "Programmation Mathématique Différentiable".

[18] . P. Crouzeix, B. Merikhi, A logarithm barrier method for semidefinite programming RAIRO-Oper.257 Res. 42.2 (2008), 123-139.

[19] Y.H. Dai and Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property. SIAM J. Optim., 10(1999) 177-182.

[20] Y.H. Dai, J.Y. Han, G.H. Liu, D.F. Sun, X. Yin and Y. Yuan, Convergence properties of nonlinear conjugate gradient methods. SIAM J. Optim. 10 (1999) 348–358.

[21] Y.H. Dai and L.Z. Liao, New conjugacy conditions and related nonlinear conjugate gradient methods. Appl. Math. Optim. 43 (2001) 87–101.

[22] G. B. Dantzig, "Maximization of a linear function of variables subject to linear inequalities", In Tj.C. Koopmans, éditeur, Activity Analysis of Production and Allocation, 339-347, Wiley, NewYork, (1951), 554-567.

[23] E.D. Dolan, J.J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program., 91(2002) 201–213.

[24] B. Fellahi, B. Merikhi, A Logarithmic Barrier Approach Via Majorant Function for Nonlinear Programming, Journal of Siberian Federal University, Mathematics and Physics 16(4) (2023) 528–539.

[25] A. V. Fiacco, G. P. Mccormick, "Nonlinear Programming : Sequential Unconstrained Minimization Techniques", Wiley, New York, (1968).

[26] R. Fletcher, Practical Methods of Optimization, 2nd Ed., J. Wiley, Sons, New York, USA (1987).

[27] R. Fletcher, C.M. Reeves, Function minimization by conjugate gradients, Comput.J., 7(1964) 149-154.

[28] K.R. Frisch, "The logarithmic potential method of convex programming", Technical report, University Institute of Economics, Oslo, Norway, (1955).

[29] J.C. Gilbert, J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, SIAM J. Optim. 2 (1) (1992) 21–42.

[30] J.C.Gilbert,Elements of Differentiable Optimization: Theory and Algorithms,Lecture notes,National School of Advanced Techniques,Paris 2007.

[31] W. Grimes, M. Achache. A path-following interior-point algorithm for monotone LCP based on a modified Newton search direction. RAIRO-Operations Research, 57(3)(2023) 1059-1073.

[32] W. Grimes, M. Achache, A. Yassine. A Newton descent logarithmic barrier interior-point algorithm for monotone LCP. RAIRO-Operations Research, 58(6)(2024) 5537-5550.

[33] M. Hamoda, M. Mamat, M. Rivaie, Z. Salleh. A conjugate gradient method with Strong Wolfe–Powell line search for unconstrained optimization. Applied Mathematical Sciences. 10 (15), (2016) 721–734 .

[34] M. R. Hestenes, E.L. Stiefel, Methods of conjugate gradients for solving linear systems, J. Res. Natl. Bur. Stand., 49(1952) 409-436.

[35] P. Huard, "Resolution of mathematical programming with nonlinear constraints by the method of centers", In J. Abadie, editor, Nonlinear Programming, North Holland, Amsterdam, The Netherlands, (1967), 207-219.

[36] Y. I. Ibrahim, H. M. Khudhur, Modified three-term conjugate gradient algorithm and its applications in image restoration, J. Electr. Eng. Comput., 28(2022) 1510-1517.

[37] N.K. Karmarkar, "A new polynomial-time algorithm for linear programming", Proc. of the 16th Annual ACM Symposium on Theory of Computing, 4, (1984), 373–395.

[38] A. Keraghel, "Analyse convexe : théorie fondamentale et exercices", Editions Dar el'Houda, Ain M'lila, Algérie, (1999).

[39] L. G. Khachiyan, "A polynomial algorithm in linear programming", Soviet Mathematics Doklady, 20, (1979), 191-194.

[40] M. Koontse, P. Kaelo, Another hybrid conjugate gradient method for unconstrained optimization, J. Nonlinear Anal. Optim., 2(2014) 127–137.

[41] A. Leulmi, S. Leulmi, Logarithmic barrier method via minorant function for linear programming. Journal of Siberian Federal University, Mathematics and Physics, 12 (2019) 191-201.

[42] A. Leulmi, B. Merikhi, D. Benterki, Study of a logarithmic barrier approach for linear semidefinite programming. Journal of Siberian Federal University, Mathematics and Physics, 11 (2018) 300-312.

[43] A. Leulmi, An efficient penalty method without a line search for nonlinear optimization. Axioms 13(3) (2024), 176.

[44] A. Leulmi, R. Ziadi, C. Souli, M. A. Saleh, A. Z. Almaymuni. An Interior Point Algorithm for Quadratic Programming Based on a New Step-Length. International Journal of Analysis and Applications, 22,(2024) 233-233.

[45] Y. Liu and C. Storey, Efficient generalized conjugate gradient algorithms,Part 1, Theory, J. Optim. Theory. Appl., 69(1991) 129-137.

[46] I.E. Livieris, V. Tampakas, P. Pintelas, A descent hybrid conjugate gradient method based on the memoryless BFGS update, Numer. Algor. 79(4) (2018) 11691185.

[47] M. Lotfi, S.M. Hosseini, An efficient hybrid conjugate gradient method with sufficient descent property for unconstrained optimization, Optim. Methods Softw. (2021).

[48] E. Mehamdia, Y. Chaib, T. Bechouat, Two modified conjugate gradient methods for unconstrained optimization and applications, RAIRO-Oper. Res., 57 (2023) 333-35.

[49] A. Mehamdia. Using conjugate gradient methods for regression function (Doctoral dissertation, university of souk ahras)(2024).

[50] P. Mtagulwa, P. Kaelo, An efficient modified PRP-FR hybrid conjugate gradient method for solving unconstrained optimization problems, Appl. Numer. Math., 145(2019) 111–120.

[51] J. Nocedal, S.J. Wright. "Numerical Optimization", Springer Series in Operations Research and Financial Engineering.

[52] M.Ouriemchi, Résolution de problèmes non linèaires par les méthodes de points intèrieurs,Théorie et algorithmes, Thèse de doctorat, Univeritè du Havre, France, 2006

[53] E. Polak, G. Ribiere, Note sur la convergence de méthodes de directions conjuguées, Revue Française d'informatique et de Recherche Opérationnelle, Série Rouge, 3(1969) 35-43 .

[54] B.T. Polyak, The conjugate gradient method in extremal problems. USSR Comput. Math. Math. Phys., 9(1969) 94–112.

[55] S.S. Rao. "Engineering Optimization Theory and Practice", JOHN WILEY and SONS, INC.

[56] M. Rivaie, M. Mamat, L. W. June, I. Mohd, A new class of nonlinear conjugate gradient coefficients with global convergence properties, Appl. Math. Comput., 218(22)(2012) 11323-11332.

[57] E. Shannon, A mathematical theory of communication, Bell Syst. Tech. J., 27(1948) 379–423 and 623–656

[58] N. Z. Shor, "Utilization of the operation of space dilatation in the minimization of convex functions", Kibernetika, 1, (1970), 6-12.

[59] C. Souli, A. Leulmi, A study of a penalty method for nonlinear optimization based on a new approximate function, journal of Nonlinear Dynamic and systems theory, 23, 5 , (2023) 561–570.

[60] C. Souli, R. Ziadi, A. Bencherif-Madani, H. M. Khudhur. A hybrid CG algorithm for nonlinear unconstrained optimization with application in image restoration. J. Math. Mod.,(2024) 301-317.

[61] C. Souli, R. Ziadi, I. Lakhdari, A. Leulmi. "An efficient hybrid conjugate gradient method for unconstrained optimization and image restoration problems." Iranian Journal of Numerical Analysis and Optimization 15, 1 (2025) 99-123.

[62] I. M. Sulaiman, N. A. Bakar, M. Mamat, B.A. Hassan, M. Malik, A. M. Ahmed, A new hybrid conjugate gradient algorithm for optimization models and its application to regression analysis J. Electr. Eng. Comput. 23(2021) 1100-1109.

[63] Wolkowicz, G.P.H. Styan, Bounds for Eigenvalues Using Traces, Lin. Alg. Appl. 29 (1980) ,471–265 506.

[64] L. Zhang, An improved Wei-Yao-Liu nonlinear conjugate gradient method for optimization computation, Appl. Math. Comput. 215(6) (2009) 2269–2274.

[65] R. Ziadi, R. Ellaia, A. Bencherif-Madani, Global optimization through a stochastic perturbation of the Polak-Ribiére conjugate gradient method, J. Comput. Appl. Math., 317,(2017) 672-684.

[66] R. Ziadi, A. Bencherif-Madani, A mixed algorithm for smooth global optimization, J. Math. Model., 11(2)(2023) 207-228.

[67] R. Ziadi, A. Bencherif-Madani, A Perturbed quasi-Newton algorithm for bound-constrained global optimization, J. Comp. Math., doi:10.4208/jcm.2307-m2023-0016

[68] G. Zoutendijk, Nonlinear programming computational methods, J. Integ. Nonlinear Progr.,(1970) 37-86.

ملخص:

في هذه الأطروحة، نركز على التحليل النظري والتحقيق العددي لبعض طرق النقطة الداخلية وطريقة التدرج المرافق لحل مسائل الأمثلة غير الخطية.

أولا، نقترح أساليب الحاجز اللوغاريتمي لمسألة الأمثلة غير خطية محدبة مع قيود، حيث يتم التعامل مع عامل الحاجز كشعاع. يلي ذلك دراسات تحليلية يتم فيها تحديد طول الخطوة باستخدام تقنية دالة الحد السفلي. تظهر النتائج العددية أن الطرق المقترحة تتميز بالكفاءة والصلابة.

بالإضافة إلى ذلك، نطور طرق تدرج مرافق جديدة لحل مسائل الأمثلة غير الخطية دون قيود. تولد هذه الطرق اتجاهات انحدار دون الحاجة إلى تقنيات البحث الخطي، علاوة على ذلك، تظهر هذه الطرق تقاربًا شاملا تحت فرضيات معتدلة. تشير النتائج العددية إلى أن الطرق المقترحة فعّالة ومرنة في معالجة مختلف مسائل الأمثلة غير الخطية غير المقيدة واستعادة الصور.

كلمات مفتاحيه: مسألة الأمثلة غير الخطية، طريقة النقطة الداخلية، الحاجز اللوغاريتمي، دالة الحد السفلي، طريقة التدرج المرافق، التقارب الشامل، اتجاه انحدار.

Résumé:

Dans cette thèse, nous nous concentrons sur l'analyse théorique et l'investigation numérique de certaines méthodes de points intérieurs et de gradient conjugué pour résoudre des problèmes d'optimisation non linéaire.

Tout d'abord, nous proposons des approches de barrière logarithmique pour les problèmes d'optimisation non linéaires convexe avec contraintes, où le paramètre de barrière est traité comme un vecteur. Cela est suivi par des études analytiques dans lesquelles le pas de déplacement est déterminé à l'aide de la technique de la fonction minorante. Les résultats numériques révèlent que les méthodes proposées démontrent à la fois efficacité et robustesse.

De plus, nous développons de nouvelles méthodes de gradient conjugué pour résoudre des problèmes d'optimisation sans contraints. Ces méthodes génèrent des directions de descente sans nécessiter de techniques de recherche linéaire. En outre, elles présentent une convergence globale sous des hypothèses. Les résultats numériques montrent que les méthodes proposées sont efficaces et robustes pour résoudre divers problèmes d'optimisation sans contraintes et de restauration d'images.

Mots clés: Un Problème d'optimisation non linéaire, Méthode de point intérieur, Barrière logarithmique, Fonction minorante, Méthode du gradient conjugué, Convergence globale, Direction de descente.

Abstract :

In this thesis, we focus on the theoretical analysis and numerical investigation of specific interior point and conjugate gradient methods for solving nonlinear optimization problems.

First, we propose logarithmic barrier approaches for constrained convex nonlinear optimization problems, where the barrier parameter is treated as a vector. This is followed by analytical studies in which the step-length is determined using the minorant function technique. The numerical findings reveal that the proposed methods exhibit both effectiveness and robustness.

In addition, we develop new conjugate gradient methods for solving unconstrained optimization problems. These methods generate descent directions without requiring line search techniques. Moreover, they exhibit global convergence under mild assumptions. Numerical results indicate that the proposed methods are both effective and robust in addressing various unconstrained optimization and image restoration problems.

Key words: Nonlinear optimization problem, Interior point method, Logarithmic barrier, Minorant function, Conjugate gradient method, Global convergence, Descent direction.