DERIVATIVE-FREE APPROACHES FOR SOLVING SYSTEM OF NONLINEAR EQUATIONS WITH APPLICATIONS

Thesis Submitted For the Award of the Degree of

DOCTOR OF PHILOSOPHY

In

(Mathematics)

By

Abubakar Sani Halilu

(Registration Number: 11918295)

Supervised By

Dr. Arunava Majumder

Co-Supervised by

Prof. Mohammed Yusuf Waziri

LOVELY PROFESSIONAL UNIVERSITY

PUNJAB

2022

DECLARATION

I, hereby declared that the presented work in the thesis entitled "**Derivative-free approaches for solving system of nonlinear equations with applications**" in fulfilment of degree of **Doctor of Philosophy (Ph.D. Mathematics)** is outcome of research work carried out by me under the supervision of Dr. Arunava Majumder, working as Assistant Professor, in the Department of Mathematics of Lovely Professional University, Punjab, India and under the co-supervision of Prof. Mohammed Yusuf Waziri from Professor, Bayero University, Kano, Nigeia. In keeping with general practice of reporting scientific observations, due acknowledgements have been made whenever work described here has been based on findings of other investigator. This work has not been submitted in part or full to any other University or Institute for the award of any degree.

Name of the scholar: Abubakar Sani Halilu

Registration No.: 11918295

Department/school: Mathematics

Lovely Professional University,

Punjab, India

CERTIFICATE

This is to certify that the work reported in the Ph.D. thesis entitled "Derivative-free approaches for solving system of nonlinear equations with applications" submitted in fulfillment of the requirement for the reward of degree of **Doctor of Philosophy (Ph.D. Mahematics)** in the Department of Mathematics, is a research work carried out by Abubakar Sani Halilu, (Registration No.: 11918295), is bona fide record of his original work carried out under my supervision and that no part of thesis has been submitted for any other degree, diploma or equivalent course.

Name of supervisor: Dr. Arunava Majumder
Designation: Assistant Professor
Department/school: Mathematics
University: Lovely Professional University.

Name of Co-Supervisor: Prof. M.Y. Waziri
Designation: Professor
Department: Mathematical Sciences
University: Bayero University, Kano.

Acknowledgment

All praise be to Allah, the Most Gracious and Merciful, for giving me life and the wisdom to pursue my Ph.D. (Mathematics) program at Lovely Professional University, Phagwara, Punjab, India. May Allah's peace and blessings be upon his messenger, Prophet Muhammad (S.A.W).

I am exceedingly grateful to my talented and ambitious supervisor, Dr. Arunava Majumder, and my professional co-supervisor, Prof. Mohammed Yusuf Waziri. They have devoted themselves to assisting and supporting me throughout my Ph.D. journey. I like to convey my heartfelt gratitude to Sule Lamido University, Kafin-Hausa, Nigeria, for providing me with the opportunity to pursue this research through the TETFund intervention. I also like to thank everyone in the Numerical Optimization Research Group for their help and contributions, especially Kabiru Ahmed, Habibu Abdullahi, Salisu Murtala, Dr. Yau Balerabe Musa, and Dr. Aliyu Muhammed Awwal.

I am incredibly grateful for my parents' encouragement, prayers, and support throughout this critical stage of my studies. My heartfelt thanks go to Saddiqa Abbas Mu'azu, my devoted wife, for her steadfast support, prayers, and patience as I pursued my degrees. I'd also like to thank my biological son, Bashir Abubakar (Halifa), and my biological daughter, Rukayya Abubakar (Mama).

My grateful appreciation goes to my beautiful friends and colleagues, whose encouragement and contributions to this success have been tremendous. Their names are far too many to list here. Their contributions will be forever remembered.

Dedication

This work, is humbly dedicated to my parents.

List of Tables

List of Figures

Contents

Abstract

In recent years, extensive applications of derivative-free methods have become a popular trend. The derivative-free methods for solving monotone systems of nonlinear equations are presented in Chapters three to eight of this thesis. The procedures in these chapters are based on the projection technique of Solodov and Svaiter. In addition, the step lengths are calculated using inexact line search technique. A new structured conjugate parameter is derived in Chapter three. In Chapter four, a new choice of the Dai-Liao nonnegative parameter t is presented. As a result, the proposed algorithm is expanded for solving a general system of nonlinear equations. A new choice for the Hager-Zhang nonnegative parameter θ_k is presented in Chapter five. In Chapter Six, the double direction methods are proposed. The first algorithm is developed by approximating the Jacobian matrix via the acceleration, while in the second algorithm, the correction parameter is presented based on the PicardMann hybrid iterative scheme proposed by Safeer (2013). In chapter seven, the proposed method improved the numerical performance of the Hager-Zhang method proposed by Waziri et al. (2019), by extending its search direction to a three-term conjugate gradient direction. In chapter eight, a modified Dai-Yuan (DY) search direction with the projection scheme, for constrained system of monotone nonlinear equations is presented. The new scheme can be seen as an adaptation of the iterative method designed by Yu and Guan (2005) for unconstrained optimization. Some nice properties of the scheme include being derivative-free and satisfying the required condition for global convergence irrespective of the line search strategy employed.

The proposed methods are proven to be globally convergent under mild conditions. The numerical experiments conducted in this work show the efficacy and performance of the proposed methods. In addition, the methods presented in Chapters three, five, six, and eight are successfully applied to handle the ℓ_1-norm regularization problem in compressive sensing, which exhibits better results than the existing methods.

CHAPTER ONE

Introduction

1.1 Introduction

This chapter shade more light on some approaches for solving a system of nonlinear equations; it contains, among other things, motivation of the study, scope and limitations, methodology, and some basic definitions.

Nonlinear problems are of interest to scientists because most problems arising from engineering, biology, mathematics, physics, and many other branches of science are inherently nonlinear. An engineer should create a design that can carry the full load with the possible lowest cost. Manufacturers desire to design their products to maximize gain and minimize cost. Scientists and other designers often look for mathematical functions that describe their data with minimum discrepancy. These are only a few instances of how optimization problems come about. One of the most significant difficulties of nonlinear problems is that it is impossible to obtain exact solutions analytically in general. As a result, iterative techniques are used as alternatives to getting approximate solutions to such problems.

The act of attaining the optimal result under given conditions is known as optimization. Engineers must make several technological and management decisions during an engineering system's design, construction, and maintenance. The end goal of all such decisions is to either minimize the required quantity or maximize the desired benefit. As a result, optimization determines the conditions that result in the optimal value of a given function. [92]. There are different approaches for solving optimization problems which include: analytical method, graphical

method, and numerical method [92]. Nonetheless, some problems have no analytical solution or are too difficult to solve, necessitating the development of numerical approaches to provide approximate solutions.

An iterative technique is a method that uses an initial estimate, say x_0, of the solution, say x^*, to generate a sequence of approximations, say $x_0, x_1, x_2, ...$ that converges in the vicinity of the solution x^*. The main objective of iterative techniques is to design methods that converge to the approximate solution from the given initial estimate as fast as possible. A convergence analysis is usually performed for every iterative algorithm developed to show if the convergence is local or global. Iterative methods are often helpful for problems with many variables where analytical methods would be prohibitively expensive.

1.2 Nonlinear System of Equations

The standard nonlinear system of equations is represented by

$$F(x) = 0, \tag{1.1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^m$ is nonlinear map, and it is assumed to satisfy the following condition:

(i) There exists an $x^* \in R^n$ s.t. $F(x^*) = 0$.

(ii) F is a continuously differentiable mapping in the neighborhood of x^*.

When $m > n$, the system is called an overdetermined and named underdetermined when $n > m$. However, if $m = n$, then the system is called a square nonlinear system of equations and is called symmetric nonlinear equations if Jacobian $F'(x)$ is symmetric. In particular, a class of nonlinear equations is a monotone nonlinear equation whenever the function F satisfies

$$\langle F(x) - F(y), x - y \rangle \geq 0, \quad \forall x, y \in \mathbb{R}^n. \tag{1.2}$$

In addition, if $x \in \Omega \subset \mathbb{R}^n$ and the set Ω is nonempty, closed and convex, then problem (1.1) is called convex constrained monotone system of nonlinear equations. That is

$$F(x) = 0, \quad x \in \Omega. \tag{1.3}$$

Minty [77], Zarantonello [128], and Kačurovskii [68], were the first to investigate monotone mappings in Hilbert spaces. The monotone nonlinear equations can be applied to solve the problems related to chemical equilibrium [76], compressive sensing [65], and economic equilibrium [101] among others.

1.3 General Unconstrained Optimization Problems

Iterative methods are very useful for solving unconstrained optimization problems of the form

$$\min f(x), \quad x \in \mathbb{R}^n, \quad f : \mathbb{R}^n \to \mathbb{R}. \tag{1.4}$$

The function $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable function at $x \in \mathbb{R}^n$. By continuously differentiable we mean $\left(\frac{\partial f}{\partial x_i}\right)(x)$ exists and continuous, $i = 1, 2, ..., n$. The function is twice continuously differentiable function at $x \in \mathbb{R}^n$ if $\left(\frac{\partial^2 f}{\partial x_i \partial x_j}\right)(x)$ exists and continuous, $j = 1, 2, ..., n$.
The gradient of f at x is defined as

$$\nabla f(x) = \left[\frac{\partial f}{\partial x_1}(x), \frac{\partial f}{\partial x_2}(x), ..., \frac{\partial f}{\partial x_n}(x)\right].$$

The Hessian of f is a square symmetric matrix and positive definite whose entries are given as

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

When $F : \mathbb{R}^n \to \mathbb{R}^n$ is the gradient mapping of some function $f : \mathbb{R}^n \to \mathbb{R}$, the nonlinear system of equations (1.1) is just the first-order necessary condition for the unconstrained optimization problem (1.4). Furthermore, suppose the function f is a merit function defined by

$$f(x) = \frac{1}{2}\|F(x)\|^2, \tag{1.5}$$

then the nonlinear equations problem (1.1) is equivalent to the global optimization problem in (1.4).

The following Theorems are very useful in unconstrained optimization problems.

Theorem 1.3.1 *When f is convex, any local minimizer x^* is a global minimizer of f. In addition if f is differentiable, then any stationary point x^* is a global minimizer of f.*

Theorem 1.3.2 *(First-Order Necessary Conditions) If x^* is a local minimizer and f is continuously differentiable in an open neighborhood of x^*, then $\nabla f(x^*) = 0$.*

Theorem 1.3.3 *(Second-Order Necessary Conditions) If x^* is a local minimizer of f and $\nabla^2 f$ exists and continuous in an open neighborhood of x^*, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semi definite.*

Theorem 1.3.4 *Suppose that $\nabla^2 f$ is continuous in an open neighborhood of x^* and that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite, then x^* is a strict local minimizer of f.*

1.3.1 Definitions of Some Basic Terms

In order to guide us through the understanding of our research, below are the definitions of some basic terms;

Definition 1.3.5 (Jacobian Matrix) *Let $F : \mathbb{R}^n \to \mathbb{R}^m$. The Jacobian matrix $F'(x)$ of $F := (F_1, F_2, \ldots, F_n)$ at $x \in \mathbb{R}^n$ is defined by*

$$F'(x) = \begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_2}{\partial x_2} & \cdots & \frac{\partial F_n}{\partial x_n} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} & \cdots & \frac{\partial F_2}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_m}{\partial x_1} & \frac{\partial F_m}{\partial x_2} & \cdots & \frac{\partial F_m}{\partial x_n} \end{bmatrix}.$$

Definition 1.3.6 (Projection) *Let $\Omega \subset \mathbb{R}^n$ be a nonempty, closed and convex set. Then for any $x \in \mathbb{R}^n$, its projection onto Ω is given by*

$$P_\Omega(x) = arg\ min\|x - y\| : y \in \Omega. \tag{1.6}$$

Definition 1.3.7 (Norm) *A Norm for an arbitrary finite-dimensional vector $x \in \mathbb{R}^n$, denoted by $\|x\|$ is a real-valued function satisfying the following four conditions for all vectors x and y of the same dimension:*

(N1): $\|x\| \geq 0$.

(N2): $\|x\| = 0$ *if and only if* $x = 0$.

(N3): $\|cx\| = |c|\|x\|$ *for any scalar c.*

(N4): $\|x+y\| \leq \|x\| + \|y\|$.

Definition 1.3.8 (Convergence) *A sequence $\{x_k\}$ of n-vectors converges to x^* if*

$$\lim_{k\to\infty} \|x_k - x^*\| = 0.$$

Definition 1.3.9 (A merit function) *This is a square norm function of the form $f(x) = \|F(x)\|^2$, where, $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ and $\|\cdot\|$ is Euclidean norm.*

Definition 1.3.10 (l_1 Norm, Euclidean Norm and l_∞ Norm) *For a vector $x \in \mathbb{R}^n$,*

- *The l_1 norm is defined as $\|x\|_1 = \sum_{i=1}^{n} (|x_i|)$.*
- *The Euclidean norm (or l_2 norm) is defined as $\|x\|_2 = \left(\sum_{i=1}^{n} \left(x_i^2\right)\right)^{\frac{1}{2}}$.*
- *The l_∞ norm is defined as $\|x\|_\infty = \max_{1\leq i\leq n} (|x_i|)$.*

Definition 1.3.11 (Local minimizer) *A point $x^* \in \mathbb{R}^n$ is is said to be a local minimizer of $f : \mathbb{R}^n \to \mathbb{R}$ if there exists a positive constant ε such that $f(x^*) \leq f(x)$ satisfying $\|x^* - x\| \leq \varepsilon$, $\forall x \in \mathbb{R}^n$.*

Definition 1.3.12 (Global minimizer) *A point $x^* \in \mathbb{R}^n$ is is said to be a global minimizer of $f : \mathbb{R}^n \to \mathbb{R}$ if $f(x^*) \leq f(x)$, $\forall x \in \mathbb{R}^n$.*

Definition 1.3.13 (Stationary point) *if x^* is a local minimizer for $f : \mathbb{R}^n \to \mathbb{R}$, then $\nabla f(x^*) = 0$.*

1.4 Nonlinear Iterative Methods

Several numerical methods arise for solving (1.4), such as steepest descent methods, Newton methods, quasi-Newton methods, and conjugate gradient method, among others. The commonly used iterative scheme for solving (1.4) is in the form

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, 2, .. \tag{1.7}$$

where x_{k+1} represents a new iterate, x_k is the previous iterate, d_k is a search direction, and α_k is a step length, to be computed using any line search rule. A line search is a technique that computes the step length α_k along the direction d_k via (1.7), by generating the sequence of iterates $\{x_k\}$ to achieve global convergence [59]. The choice of the step length affect both the convergence and convergence speed of the algorithm. In order to calculate the step length α_k, either exact or inexact procedure can be used. The exact line search rule is the ideal line search rule [96] that can be computed via

$$f(x_k + \alpha_k d_k) = \min_{\alpha > 0} f(x_k + \alpha d_k). \tag{1.8}$$

However, finding the correct step length in a practical computation is difficult, if not impossible. As a result, the most widely used line search is an inexact one [69, 40, 107], that effectively reduces the function values i.e $f(x_k + \alpha_k d_k) \leq f(x_k)$. In most of the nonlinear problems, the search direction, d_k, must be controlled with some parameter $\alpha_k > 0$, called the step length. In addition, the search direction d_k is usually necessary to meet either descent condition

$$\nabla f(x)^T d_k < 0,$$

or sufficient descent condition

$$\nabla f(x)^T d_k \leq -c\|\nabla f(x)\|^2, \quad c > 0,$$

which are very crucial in proving global convergence.

1.4.1 Newton's Method

The Newton's method, which generates iterative sequences x_k from a given starting point x_0 in the vicinity of x^* via (1.7), with search direction given by

$$d_k = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k), \tag{1.9}$$

is the most well-known approach for finding the solution to (1.4). Here $k = 0, 1, 2, \ldots$ and $\nabla^2 f(x_k)$ is the Hessian matrix of f at x_k. Newton's method stands out because of its nice properties, such as rapid convergence rate from a reasonably good starting point [43]. However, in Newton's method, the Hessian matrix can be computed at each iteration, which may be unavailable or could not be obtained exactly. As a result, Newton's approach cannot be used directly [40, 107, 110].

1.4.2 Fixed Newton Method

Fixed Newton method is the modified Newton's method. The method that eliminates computing and storing the Hessian matrix (except for k=0) [81]. Its search direction is defined by

$$d_k = -(\nabla^2 f(x_0))^{-1} \nabla f(x_k), \tag{1.10}$$

where $k = 0, 1, 2, \ldots$. However, it still involves the solution of a system of n linear equations, which requires more CPU time as the system's dimension expands [107].

1.4.3 Quasi-Newton Method

Quasi-Newton methods were developed to replace the Hessian matrix or its inverse with an approximation that may be updated at each iteration [75, 91], and there search direction is given by

$$d_k = -B_k^{-1} \nabla f(x), \tag{1.11}$$

where B_k is the approximation of Hessian matrix of f at x_k. The reasoning for quasi-Newton approaches is related to the Hessian matrix evaluation cost [21, 75, 55]. There are many formulae for updating $\{B_k\}$ but the most popular ones are BFGS (Broyden, Fletcher, Goldfarb, and Shanno) [54] and symmetric-rank-one (SR1) [39] respectively defined by

- $B_{k+1} = B_k - \dfrac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \dfrac{y_k y_k^T}{y_k^T s_k}$.
- $B_{k+1} = B_k - \dfrac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}$.

Furthermore, B_k needs to satisfy the secant equation

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

where, $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.
The two updates generate symmetric matrices. If the initial approximation $\{B_0\}$ is positive definite and $s_k^T y_k > 0$, then BFGS update generates positive definite Jacobian approximations. However, SR1 update does not guarantee positive definiteness.

Despite the appealing characteristics of the Newton and quasi-Newton's methods, the derivative F' or its approximation is computed at each iteration, so they are not ideal for solving large-scale problems because of their requirement for huge matrix storage at each iteration, which is costly in numerical experiments. To overcome these problems, matrix-free methods are proposed.

Conjugate Gradient Methods

Conjugate gradient (CG) methods are very popular and promising for large-scale nonlinear problems due to their simplicity and low storage requirements, because they neither form nor store matrices throughout the iteration process. Given a starting point $x_0 \in R^n$, nonlinear CG methods generate a sequence of iterates $\{x_k\}$ using (1.7) where the search direction d_k is generated by the following rule

$$d_k = \begin{cases} -\nabla f(x_k), & \text{if} \quad k = 0, \\ -\nabla f(x_k) + \beta_k d_{k-1}, & \text{if} \quad k \geq 1, \end{cases} \tag{1.12}$$

where, β_k is a scalar describing the characteristics of the CG methods. The rationale behind any CG method is to produce and effective conjugate gradient parameter β_k. There are many formulae for calculating the CG parameter β_k and different CG methods correspond to different choices for the scalar β_k. Below are some of the well known CG parameters:

- $\beta_k^{FR} = \dfrac{\|\nabla f(x_k)\|^2}{\|\nabla f(x_{k-1})\|^2}$ (Fletcher-Reeves (FR) [50]).
- $\beta_k^{DY} = \dfrac{\|\nabla f(x_k)\|^2}{d_{k-1}^T y_{k-1}}$ (Dai-Yuan (DY) [35]).
- $\beta_k^{CD} = \dfrac{\|\nabla f(x_k)\|^2}{-d_{k-1}^T \nabla f(x_{k-1})}$ (Fletcher (conjugate descent (CD)) [49]).
- $\beta_k^{PRP} = \dfrac{\nabla f(x_k)_k^T y_{k-1}}{\|\nabla f(x_{k-1})\|^2}$ (Polak-Ribi*é*re-Polyak (PRP) [88]).
- $\beta_k^{HS} = \dfrac{\nabla f(x_k)^T y_{k-1}}{d_{k-1}^T y_{k-1}}$ (Hestenes-Stiefel (HS) [66]).
- $\beta_k^{LS} = \dfrac{\nabla f(x_k)^T y_{k-1}}{-d_{k-1}^T \nabla f(x_{k-1})}$ (Liu and Storey (LS) [74]).

where $\|\cdot\|$ is the Euclidean norm. It can be observed that whenever $\beta_k = 0$, then the CG method return to the well-known steepest descent method to compute its search direction.

1.4.4 Spectral Gradient Methods

Spectral gradient (SG) methods are very suitable for large-scale unconstrained optimization problems. The first spectral gradient method was introduced by Barzilai and Borwein [20] for unconstrained optimization problems and the idea was first incorporated to solve large-scale nonlinear system of equations by La Cruz and Raydan [28]. An attractive property of these methods is that they are globally convergence and very easy to implement. They however generate sequence of

iterates $\{x_k\}$ via (1.7), with the search direction d_k is defined as

$$d_k = \begin{cases} -\nabla f(x_k), & \text{if} \quad k = 0, \\ -\lambda_k \nabla f(x_k), & \text{if} \quad k \geq 1, \end{cases} \tag{1.13}$$

where λ_k is called the SG parameter. It is very important for the scalar parameter λ_k to be strictly positive. It can be observed that if $\lambda_k = 1$, then the SG method reduces to the well-known steepest descent method. Different choices of spectral parameterλ_k correspond to different spectral gradient methods.

1.4.5 Spectral Conjugate Gradient Methods

Spectral gradient methods may make the iterate points zigzag which have similar disadvantage to that of the steepest descent method [72]. To overcome this drawback, it make sense to combine the spectral parameter and conjugate gradient parameter to defined a formula that generate search direction d_k via

$$d_k = \begin{cases} -\nabla f(x_k), & \text{if} \quad k = 0, \\ -\lambda_k \nabla f(x_k) + \beta_k d_{k-1}, & \text{if} \quad k \geq 1. \end{cases} \tag{1.14}$$

Iterative algorithms that generate search direction using (1.14) are called spectral conjugate gradient (SCG) methods. It can be observed that if $\lambda_k = 1 \ \forall k > 0$, then the SCG direction (1.14) reduces to the CG direction (1.12) and if $\beta_k = 1 \ \forall k > 0$, then (1.14) reduces to the SG direction (1.13).

1.5 Statement of the Problem

Newton methods and their improved version, i.e., the quasi-Newton methods for solving system of nonlinear equations require the computation and storage of the Jacobian matrix or it's approximation at each iteration. Moreover, the convergence of some Newton methods or their variant is very poor due to less information at a point or singularity of the matrix at a given point. Therefore, the proposed

methods are derivative-free and singularity-free methods with less computational cost compared to some existing methods.

1.6 Motivation of the Study

The study of derivative-free and low storage requirement methods are mainly developed for unconstrained optimization problems,due to there low storage requirement. In particular, most of the conjugate gradient directions do not necessarily generate decent conditions, which is the very important factor in achieving the global convergence. However, the study of derivative-free methods for solving system of nonlinear equations are very scanty in the existing literature. Therefore, we are motivated to suggest some derivative-free and singularity-free methods with less computational cost that are globally convergent.

1.7 Objectives

The objectives of this research are:

1. To propose structured conjugate gradient methods for solving monotone nonlinear equations.

2. To prove the global convergence of the proposed algorithms and present comprehensive numerical experiment of the methods by making comparison with some existing methods in the literature.

3. To apply some of the proposed derivative-free methods to solved problems arising in compressive sensing.

1.8 Scope and Limitations

This research is only limited to nonlinear system of equations consisting of n-equations and n-variables.

The thesis is organized as follows. In chapter two, literature review will be presented. The proposed methods' algorithms will be detailed in Chapters three through eight. In chapter nine, summary and conclusions will be presented.

CHAPTER TWO

Literature Review

In this chapter, we present review of the related literature more especially those related to unconstrained optimization problems and system of nonlinear equations. We however present the literature related to monotone nonlinear equations and their applications to solve ℓ_1-norm regularization problem in compressive sensing.

2.1 Existing Works on Derivative-free Methods

Despite the appealing characteristics of the Newton and quasi-Newton's methods, the derivative F' or its approximation is computed at each iteration, so they are not ideal for solving large-scale problems because of their requirement for huge matrix storage at each iteration, especially when the dimension is increasing which is costly in numerical experiments. To overcome these problems, matrix-free methods are proposed. Barzilai and Borwein [20] is among the successful matrix-free methods that generates a sequence of iterates as $x_{k+1} = x_k - \tau_k F_k$ and τ_k is the step length, which can be written as $x_{k+1} = x_k - D_k F_k$, where $D_k = \tau_k I$ which is supposed to satisfy the secant equation (2.30). The step length τ_k can be obtained by minimizing the least square problems $\min_{\tau} \|\tau s_{k-1} - y_{k-1}\|^2$ and $\min_{\tau} \|s_{k-1} - \tau y_{k-1}\|^2$. These yield

$$\tau_k^{BB1} = \frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \quad \text{and} \quad \tau_k^{BB2} = \frac{s_{k-1}^T y_{k-1}}{\|y_{k-1}\|^2}. \tag{2.15}$$

respectively. Mohammed in [79] proposed Barzilai-Borwein-like method by approximating the Jacobian matrix with $\bar{\tau}_k I$, where

$$\bar{\tau}_k = \min\left\{\max\left\{\frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}}, \bar{\tau}_{\min}\right\}, \bar{\tau}_{\max}\right\}.$$

Consequently, the author in [79] used Barzilai-Borwein (BB) like method and approximating the matrix B_k^{-1} in quasi-Newton with diagonal matrix (i.e. $B_k^{-1} \approx \tau_k^{BB1} I$), and the search direction is given by $d_k = -\tau_k^{BB1} F(x_k)$, which gives the method advantage to solve large-scale problems. In [60], Halilu et al. approximate the gradient $\nabla f(x_k)$ via $g_k \approx \nabla f(x_k)$, where $f(x) = \frac{1}{2}\|F(x)\|^2$, and

$$g_k = \frac{F(x_k + \alpha_k F(x_k)) - F(x_k)}{\alpha_k}.$$

This avoids computing the exact gradient and keeping α_k updated using the line search method. However, the search direction is given by $d_k = -\gamma_k^{-1} F(x_k)$, where γ_k is a positive parameter that can be updated as

$$\gamma_{k+1} = \frac{y_k^T [F(x_k + \alpha_k F(x_k)) - F(x_k)]}{s_k^T [F(x_k + \alpha_k F(x_k)) - F(x_k)]}.$$

Another variant of matrix-free approaches are the double direction methods. In 2014, Petrović and Stanimirović [86] presented the double direction scheme for unconstrained optimization problems. In general the double direction scheme generated an iterative sequence as

$$x_{k+1} = x_k + \alpha_k d_k^i + \alpha_k^2 d_k^{ii}, \tag{2.16}$$

where x_{k+1} represents the new iterate, x_k is the previous iterate, α_k denotes the step length, while d_k^i and d_k^{ii} are the search directions. In [86], An acceleration parameter γ_k is used to approximate the Hessian matrix with a diagonal matrix (i.e., $\nabla^2 f(x_k) \approx \gamma_k I$), where $f : \mathbb{R}^n \to \mathbb{R}$. Consequently, Petrović [84], extended the work in [86] and proposed double step length scheme given by

$$x_{k+1} = x_k + \alpha_k d_k + \beta_k c_k, \tag{2.17}$$

where α_k and β_k are the respective step lengths, while d_k and c_k are the search directions. The global convergence was established under the assumption that the gradient of the function f is Lipschitz continuous. According to the numerical results the proposed method [84] is more efficient than the double direction method [86]. Subsequently, to enhance the numerical results of double step length [84], Stanimirović et al. [98] transformed the double step length scheme with the additional assumption that $\alpha_k + \beta_k = 1$. To improve the convergence properties of the double direction method presented in [86], it is hybrid version has been proposed in [87], this is made possible by hybridizing the double direction scheme with Picard-Mann hybrid iterative process proposed by Khan in [95], where the Picard-Mann hybrid iterative process is defined as three relations:

$$\begin{cases} x_1 = x \in \mathbb{R}^n, \\ v_k = (1-\eta_k)x_k + \eta_k T(x_k), \\ x_{k+1} = T(v_k), \quad k \in \mathbb{N}, \end{cases} \tag{2.18}$$

where $T : \Omega \longrightarrow \Omega$ is a mapping defined on nonempty convex subset Ω of a normed space $\mathbf{E}$, x_k and v_k are sequences determined by the iteration (2.18), and $\{\eta_k\}$ is the sequence of positive numbers in $(0,1)$. Furthermore, Petrović combined the idea presented in [84] with the iterative scheme proposed by Khan [95], and presented the hybridization method applied on a double step length [85].

On the other hand, there is very little research on double direction methods for solving system of nonlinear equations in the current literature. For this reason, Halilu and Waziri became interested and proposed the double direction method for solving the system of nonlinear equations [59]. They used the concept of a double direction scheme in their work [59] by approximating the Jacobian matrix with a diagonal matrix ($F'_k \approx \eta_k I$) via the acceleration parameter $\eta_k > 0$. The two directions in the scheme are presented as $d_k^i = -\eta_k^{-1} F(x_k)$ and $d_k^{ii} = -F(x_k)$. The convergence analysis of the proposed algorithm was established under the basic assumptions. The numerical results reported in [59] demonstrated that the proposed method outperformed the single direction method [133] in the existing literature. Following that, Halilu and Waziri [62] expanded on the work in [59] to improve its numerical performance. This is accomplished by making the two

directions in the double direction approach equal (i.e., $d_k^i = d_k^{ii} = -\eta_k^{-1}F(x_k)$. The numerical results demonstrated that the presented method [62] is efficient because it has fewer iterations and requires less CPU time than the compared method [133]. Motivated by the double step length method for unconstrained optimization [84], Halilu and Waziri incorporated the idea and proposed inexact double step length method for solving system of nonlinear equations [63]. The proposed method generated a sequence of iterations $\{x_k\}$ such that $x_{k+1} = x_k + (\alpha_k + \beta_k)d_k$, where α_k and β_k are two step lengths computed using inexact line search proposed by Li and Fukushima [69]. Following that, the same authors transformed the double direction scheme based on the assumptions that $\beta_k = \frac{1}{2}\alpha_k$ [58]. However, using the inexact line search proposed in [69], the method is shown to be globally convergent. For further research in the double step length scheme, a modification of the double step length method [63] has recently been presented in [64], the method generated the iterations as $x_{k+1} = x_k - (\alpha_k + \beta_k\lambda_k)\lambda_k^{-1}F(x_k)$, where $\lambda_k > 0$ is an acceleration parameter that is updated at each iteration as

$$\lambda_{k+1} = \frac{y_k^T y_k}{(\alpha_k + \beta_k\lambda_k)y_k^T d_k}.$$

The numerical results have shown that the proposed method outperform the compared method [63], because it converges faster.

The logic behind the double direction and step length methods is that each iteration contains two corrections; if one fails during the iteration process, the other will automatically correct the system.

Conjugate gradient methods are another type of derivative-free approach. These methods are effective for dealing with large-scale problems. The CG method for unconstrained optimization problems generate the sequence of iterate via (1.7), with the CG search direction d_k defined in (1.12). The reasoning behind using $d_k = -\nabla f(x_k)$, if $k = 0$ in a CG algorithm is that, Crowder and Wolfe [26] provided a 3-dimensional example in 1972 demonstrated that if the initial search direction is not steepest descent direction, then the convergence rate is linear. It can be observed that, FR [50], DY [35], and CD [49] methods have numerator $\|\nabla f(x_k)\|^2$. Theoretically, these methods differ from others because their global convergence theorems rely solely on the Lipschitz assumption, rather than the

boundedness assumption. Despite the fact that the methods have certain good convergence properties, Powell [89] noted that, if a small step is generated from the solution point, their numerical results are defined as weaker; subsequent steps may also be very short. This is referred to as the jamming phenomenon. On the other hand, *HS* [66], *PRP* [88], and *LS* [74] methods with numerator $\nabla f(x_k)_k^T y_{k-1}$, have adequate numerical performance, since they have a built-in restart feature for dealing with jamming problem of *CD*, *FR* and *DY* methods. Furthermore, they outperform the *CD*, *FR* and *DY* methods because y_{k-1} in the numerator of the CG parameter approaches zero if $x_k - x_{k-1}$ is too small. Hence, search direction d_k is effectively reverted to the steepest descent direction. As a result, they can automatically recover in order to avoid jamming. However, the their convergence properties are inefficient. Because, in the absence of restarts, the *LS*, *PRP* and *HS* methods can circle indefinitely without approaching a solution Powell [90]. As a result, considerable effort has been expended over the years in developing new formulae for globally convergent, robust, and numerically efficient CG methods.

Wei et al. [129] proposed an updated *PRP* method for unconstrained optimization with the CG parameter defined by

$$\beta_k^{WYL} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}\nabla f(x_k)^T \nabla f(x_{k-1})}{\|\nabla f(x_{k-1})\|^2}. \tag{2.19}$$

One of the nice features of this method is that the β_k^{WYL} is never negative in any iteration, allowing it to avoid jamming. Zhang [123] further revised the *WYL CG* parameter by replacing $\nabla f(x_k)^T \nabla f(x_{k-1})$ with $|\nabla f(x_k)^T \nabla f(x_{k-1})|$ and proposed another CG method called *NPRP* to obtain some nice convergence property. As a result, Dai and Wen [38] looked at the work of Wei et al. [129] and suggested the *DPRP* approach. The β_k^{DPRP} formula is as follows:

$$\beta_k^{DPRP} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}|\nabla f(x_k)^T \nabla f(x_{k-1})|}{\mu|\nabla f(x_k)^T d_{k-1}| + \|\nabla f(x_{k-1})\|^2}, \quad \mu > 1. \tag{2.20}$$

Sun in [100], later updated the *DPRP* method in [38] and introduced another CG parameter as:

$$\beta_k^{MPRP} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}\max\{\nabla f(x_k)^T \nabla f(x_{k-1}), 0\}}{\mu|\nabla f(x_k)^T d_{k-1}| + \|\nabla f(x_{k-1})\|^2}, \quad \mu \geq 0. \quad (2.21)$$

To improve the convergence properties of *HS* and *LS* methods, *DHS* and *DLS* methods are proposed and defined as follows:

$$\beta_k^{DHS} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}|\nabla f(x_k)^T \nabla f(x_{k-1})|}{\mu|\nabla f(x_k)^T d_{k-1}| + d_{k-1}^T y_{k-1}}, \text{ (Dai and Wen [33]).}$$

$$\beta_k^{DLS} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}|\nabla f(x_k)^T \nabla f(x_{k-1})|}{\mu|\nabla f(x_k)^T d_{k-1}| - d_{k-1}^T \nabla f(x_k)}, \text{ (Zheng and Zheng [131]).}$$

To address the shortcomings of the classical CG methods, hybrid CG methods and parametric families have been proposed. As the name implies, are based on combining the two different methods in order to present an effective method with better numerical performance and convergence properties. The following hybrid approach was proposed by Ahmad and Storey [103]:

$$\beta_k = \begin{cases} \beta_k^{PRP} & if \quad 0 \leq \beta_k^{PRP} \leq \beta_k^{FR}, \\ \beta_k^{FR} & \text{Otherwise.} \end{cases} \quad (2.22)$$

When an iteration jam occurs again, the *PRP* update parameter is used. Consequently, Hu and Storey [67] proposed that β_k be defined as:

$$\beta_k = \max\{0, \min\{\beta_k^{PRP}, \beta_k^{FR}\}\}, \quad (2.23)$$

According to Nocedal and Gilbert [51] that β_k^{PRP} can be negative. Consequently, they proposed the following CG parameter

$$\beta_k = \max\{-\beta_k^{FR}, \min\{\beta_k^{PRP}, \beta_k^{FR}\}\}, \quad (2.24)$$

in an attempt to broaden the available options for the PRP update parameter. In [36], Dai and Yuan used the DY parameter in conjunction with HS parameter to present the following CG parameter:

$$\beta_k = \max\{0, \min\{\beta_k^{DY}, \beta_k^{HS}\}\}, \tag{2.25}$$

and

$$\beta_k = \max\{-c\beta_k^{DY}, \min\{\beta_k^{DY}, \beta_k^{HS}\}\}, \quad c = \frac{1-\sigma}{1+\sigma} > 0. \tag{2.26}$$

where $0 < \sigma < 1$.

Dai and Ni investigated various CG methods for large-scale unconstrained optimization problems in [32] and found that the hybrid CG approach (2.25) produced the best results.

To enhance the numerical results and convergence properties of the CG methods, one or more parameters may be introduced to combine methods. Gonglin [53] proposed a hybrid method for unconstrained optimization based on a linear combination of the classical FR method and the WYL CG method in [129], which is described as:

$$\beta_k^H = \lambda_1 \beta_k^{WYL} + \lambda_2 \beta_k^{FR}, \tag{2.27}$$

where λ_1 and λ_2 are non zero real numbers.

Liu and Li introduced a new CG parameter in [71] by combining the LS and DY parameters as a convex combination in the following way:

$$\beta_k = (1-\theta_k)\beta_k^{LS} + \theta_k \beta_k^{DY}, \tag{2.28}$$

where $\theta_k \in [0,1]$ is the hybridization parameter. It can be seen that When $\theta_k = 0$, $\beta_k = \beta_k^{LS}$, and when $\theta_k = 1$, $\beta_k = \beta_k^{DY}$.

In [34, 37], Dai and Yuan presented a family of CG approaches with the CG parameter given as:

$$\beta_k = \frac{\|\nabla f(x_k)\|^2}{\theta_k \|\nabla f(x_{k-1})\|^2 + (1-\theta_k) d_{k-1}^T y_{k-1}}, \tag{2.29}$$

where $\theta_k \in [0,1]$. Note that, If $\theta_k = 0$, then $\beta_k = \beta_k^{DY}$, and if $\theta_k = 1$, then $\beta_k = \beta_k^{FR}$.

Researchers have sought to design CG methods that are not only globally convergent but also numerically effective because certain CG approaches for unconstrained optimization are not globally convergent. In quasi-Newton methods, the search direction d_k is needed to satisfy the secant equation given as

$$y_{k-1} = B_k s_{k-1}, \tag{2.30}$$

where, $s_{k-1} = x_k - x_{k-1}$ and $y_k = \nabla f(x_k) - \nabla f(x_{k-1})$.
The conjugacy condition for nonlinear conjugate gradient methods is given by

$$d_k^T y_{k-1} = 0. \tag{2.31}$$

Based on this requirement, by exploiting the following quasi-Newton search direction

$$B_k d_k = -\nabla f(x_k), \tag{2.32}$$

From (2.30) and (2.32), the following relation is obtained:

$$d_k^T y_{k-1} = d_k^T (B_k s_{k-1}) = (B_k d_k)^T s_{k-1} = -\nabla f(x_k)^T s_{k-1},$$

where B_k is a symmetric matrix, which approximates the Hessian matrix. Perry in [83] extended condition in (2.31) as

$$d_k^T y_{k-1} = -\nabla f(x_k)^T s_{k-1}. \tag{2.33}$$

In order to improve the Perry's conjugacy condition in (2.33), Dai and Liao [31] proposed a new conjugacy condition with nonnegative parameter t given as

$$d_k^T y_{k-1} = -t \nabla f(x_k)^T s_{k-1}. \tag{2.34}$$

By substituting (2.34) into the CG direction in (1.12), the Dai-Liao (DL) parameter is proposed as

$$\beta_k^{DL} = \frac{(y_{k-1} - t s_{k-1})^T \nabla f(x_k)}{d_{k-1}^T y_{k-1}}, \qquad t \geq 0. \tag{2.35}$$

Numerical results have shown that the DL method is efficient, however it depends greatly on the parameter $t \geq 0$, which its optimal value has been the subject of research over the years [113, 114, 15, 16, 18]. This follows the open problem put forward by Andrei [8]. Interestingly, the CG parameter in (2.35) reduces to some essential CG methods for certain values of the parameter t. For example, for $t = 0$, (2.35) reduces to the HS method [66]. Furthermore, despite the numerical efficiency of the DL method, its search direction does not satisfy the descent condition. As a result, researchers proposed modified variants of the DL method after being inspired by it. Here are a few examples of these methods:

$$\beta_k^{DHSDL} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}|\nabla f(x_k)^T \nabla f(x_{k-1})|}{\mu|\nabla f(x_k)^T d_{k-1}| + d_{k-1}^T y_{k-1}} - t\frac{\nabla f(x_k)^T s_{k-1}}{d_{k-1}^T y_{k-1}}, \text{ (Zheng and Zheng [131])}.$$

$$\beta_k^{DLSDL} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}|\nabla f(x_k)^T \nabla f(x_{k-1})|}{\mu|\nabla f(x_k)^T d_{k-1}| - d_{k-1}^T \nabla f(x_k)} - t\frac{\nabla f(x_k)^T s_{k-1}}{d_{k-1}^T y_{k-1}}, \text{ (Zheng and Zheng [131])}.$$

$$\beta_k^{MHSDL} = \frac{\|\nabla f(x_k)\|^2 - \frac{\|\nabla f(x_k)\|}{\|\nabla f(x_{k-1})\|}\nabla f(x_k)^T \nabla f(x_{k-1})}{d_{k-1}^T y_{k-1}} - t\frac{\nabla f(x_k)^T s_{k-1}}{d_{k-1}^T y_{k-1}}, \text{ (Yao et al. [122])}.$$

Also, the method designed by Hager and Zhang [56], which is defined as

$$\beta_k^{HZ} = \frac{\nabla f(x_k)^T y_{k-1}}{d_{k-1}^T y_{k-1}} - 2\frac{\|y_{k-1}\|^2 \nabla f(x_k)^T d_{k-1}}{(d_{k-1}^T y_{k-1})^2}, \tag{2.36}$$

represents a special case of β_k^{DL} with the parameter $t = 2\frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}}$. It has been shown that the search direction developed with (2.36) satisfies the sufficient descent condition $\nabla f(x_k)^T d_k = -\frac{7}{8}\|\nabla f(x_k)\|^2$. In [41], Delladji et al. proposed a CG method by combining β_k^{HZ} and β_k^{PRP} in the following way

$$\beta_k^{hPRPHZ} = (1-\theta_k)\beta_k^{HZ} + \theta_k \beta_k^{PRP}, \tag{2.37}$$

where $\theta_k \in [0,1]$ is the hybridization parameter. It is clear that When $\theta_k = 0$, $\beta_k^{hPRPHZ} = \beta_k^{HZ}$, and when $\theta_k = 1$, $\beta_k^{hPRPHZ} = \beta_k^{PRP}$. Another version of the DL method has been proposed by Dai and Kou [30] with the CG parameter given as

$$\beta_k^{DK} = \frac{\nabla f(x_k)^T y_{k-1}}{d_{k-1}^T y_{k-1}} - \left(\psi_k + \frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}} - \frac{s_{k-1}^T y_{k-1}}{\|s_{k-1}\|^2} \right) \frac{\nabla f(x_k)^T s_{k-1}}{d_{k-1}^T y_{k-1}},$$

where ψ_k corresponds to the scaling parameter of the scaled memoryless BFGS scheme. Clearly, β_k^{DK} is an adaptive version of β_k^{DL} with $t = \psi_k + \frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}} - \frac{s_{k-1}^T y_{k-1}}{\|s_{k-1}\|^2}$. In recent years, researchers have developed interest in finding appropriate values for DL parameter t. Babaie-Kafaki and Ghanbari [14] proposed two adaptive choices for t based on the DL method's singular value analysis, given as

$$t_1 = \frac{s_{k-1}^T y_{k-1}}{\|s_{k-1}\|^2} + \frac{\|y_{k-1}\|}{\|s_{k-1}\|} \quad \text{and} \quad t_2 = \frac{\|y_{k-1}\|}{\|s_{k-1}\|}.$$

The numerical results recorded in [14] showed efficacy compared to the methods presented in[56] and [30]. Waziri et al. [113] took the concept from [31] and presented a DL method for solving system of nonlinear equations. For further research and study on other approximations of the DL parameter, the reader may refer to [15, 16, 18, 9, 10, 45, 46, 47].

2.1.1 Existing Works on Monotone Nonlinear Equations

In recent years, some researchers have been inspired by the projection technique proposed by Solodov and Svaiter [97] to incorporate the concept of spectral gradient methods, conjugate gradient methods, and spectral conjugate gradient methods for solving unconstrained optimization problems (1.4) to solve monotone equations (1.3). For example, Li et al. [70] used the modified *PRP* approaches for unconstrained optimization problems presented in [132, 23] to solve monotone nonlinear equations using the projection technique [97]. Similarly, Cheng [25] merged the classic *PRP* method with the hyperplane projection in [97] to solve a system of monotone equations. In a numerical comparison with another method proposed by Zhang and Zhou in [130], the numerical results showed that the

method in [25] performed better. Yan et al. [121] also proposed two derivative-free methods for solving large-scale nonlinear monotone equations. First method is an expansion of two term CG method introduced in [23] for unconstrained optimization problems with minor modifications, whereas the other is based on the three term CG method presented in [132]. Under mild conditions, the methods proved to be converged globally. Cruz in [27] and Cruz and Raydan in [117] used the spectral gradient approach for unconstrained optimization, that was proposed by Barzilai and Borwein in [20]. Subsequently, Zhang and Zhou [132] used the concept in the spectral gradient approach in [20] to solve nonlinear monotone equations based on the projection method in [97]. The monotonicity of F is taken into account by introducing a new line search technique. Moreover, Liu and Li [72] proposed the spectral conjugate gradient approach, which was inspired by work presented in [35] and [97]. They also demonstrated the method's efficiency by establishing global convergence and presenting preliminary numerical experiments. Abubakar and Kumam [2], modified the work in [31] by presenting a decent DL method for monotone nonlinear equations. Waziri et al. [112] proposed a family of Hager-Zhang CG methods for monotone nonlinear equations, inspired by the unconstrained optimization problems in [57] and the hyperplane technique in [97]. Using eigenvalue analysis, the authors demonstrated that the search directions generated by the schemes were in fact, descent directions. Following that, Waziri et al. [114] also used the idea in [31] and presented the two decent search directions. Recently, Sabi'u et al. [94] improved on the work in [112] by obtaining other choices for the Hager-Zhang parameter in [57], which are employed to develop other versions of the scheme in [112].

On the other hand, many methods to solve constrained monotone nonlinear equations have been developed by some researchers, inspired by the projection method of Solodov and Svaiter [97]. Wang et al. [104], for example, used the approach in [97] to solve monotone nonlinear equations with convex constraints. After initialization, the linear system of equations is solved to obtain a point of trial, and the projection method is then used to achieve the next iteration. The method has been shown to be globally convergent with a linear rate of convergence. As a result, Wang and Wang [105] developed a modification of the method in [104] with superlinear rate of convergence. Yu et al. [127] extended Zhang

and Zhou's spectral gradient method to solve monotone nonlinear equations with convex constraints. In [120], Xiao and Zhu brought the idea proposed by Hager-Zhang in [56] and solved monotone nonlinear equations. Following that, Liu and Li[73], provided the CG parameter by revising the work of Xiao and Zhu [120]. Mohammed and Abubakar [79] developed a derivative-fee decent method for monotone nonlinear equations by incorporating the work of Barzilai-Borwein [20]. Recently, Sabi'u et al. [93] extended the Hager-Zhang scheme to solving system of monotone nonlinear equations with convex constraint, by presenting two other choices for the Hager-Zhang parameter using singular value analysis.

2.2 Description of ℓ_1-norm regularization problem in compressive sensing

Many practical problems in science and technology involve the task of inferring quantities of interest from measured data. For example, in image and signal processing, it is desirable to reconstruct a signal from measured data.

Compressive sensing (CS) is essentially a method for efficiently acquiring and reconstructing a signal. It is widely used in medical science, biological engineering, as well as other fields of science and engineering [19, 22]. It compresses the signal obtained during sensing.

The CS problem is concerned with recovering a sparse signal x, from the linear system shown below.

$$\vartheta = Mx, \tag{2.38}$$

where $x \in \mathbb{R}^n, \vartheta \in \mathbb{R}^m, M \in \mathbb{R}^{m\times n}(m << n)$ denotes a linear operator (information). By solving the above linear system, one tries to recover the vector $x \in \mathbb{R}^n$. According to conventional wisdom, the number m of measurements (amount of measured data), must be at least as large as the signal length n (number of x components). In fact, if $m < n$, then the linear system (2.38) is underdetermined and has an infinite number of solutions (as long as at least one exists). In other words, without additional information, it is impossible to recover x from ϑ when $m < n$. This is also connected to the Shannon sampling theorem, which states that the

sampling rate of a continuous-time signal must be twice its highest frequency in order to ensure reconstruction.

Consequently, it was surprising that, when the number m of available measurements is less than the signal length n, it is possible to reconstruct signals under certain conditions. Even more surprisingly, efficient reconstruction algorithms do exist. Sparsity is the underlying assumption that allows all of this to happen. A signal is said to be sparse if the majority of its components are zero. Many real-world signals, as objectively recorded, are compressible in the sense that they can be well approximated by sparse signals with a suitable change of basis. The name given to the field of study that deals with this phenomenon are called *sparse recovery, compressed sensing, compressive sampling, or compressive sensing*. If the measurement vector ϑ is obtained from a highly sparse signal, the sparsest solution among all possible solutions should be sought.

The most well-known method for finding the sparse solution is to minimize an objective function as shown below.

$$\min_x \frac{1}{2}\|\vartheta - Mx\|_2^2 + \pi\|x\|_1, \tag{2.39}$$

where π it is a regularized positive parameter. In the literature, (2.39) is frequently referred to as the ℓ_1 regularized least square problem. Because ℓ_1-norm is a nonsmooth function, algorithms that need differentiability assumption are inapplicable to problem (2.39). As a result, $\|x\|_1$ is either approximated with some smooth functions or the problem (2.39) is reformulated into an equivalent problem. The gradient projection for sparse reconstruction (GPSR) method proposed by Figueiredo et al. [48] is among the effective methods for solving (2.39). They write the following convex quadratic program to solve problem (2.39). Any vector $x \in \mathbb{R}^n$ is divided into positive and negative components as follows:

$$x = \phi - \psi, \quad \phi \geq 0, \quad \psi \geq 0, \quad \phi, \psi \in \mathbb{R}^n. \tag{2.40}$$

Let $\phi_i = (x_i)_+$, and $\psi_i = (-x_i)_+$ for $i = 1,2,...,n$, where $(.)_+$ is the positive operator, which is defined as $(x)_+ = \max\{0,x\}$. Applying the definition of the ℓ_1-norm, we have $\|x\|_1 = e_n^T\phi + e_n^T\psi$, with $e_n = (1,1,1,...,1)^T \in \mathbb{R}^n$. Therefore,

problem (2.39) can be rewritten as

$$\min_{\phi,\psi} \frac{1}{2}\|\vartheta - M(\phi-\psi)\|_2^2 + \pi e_n^T\phi + \pi e_n^T\psi, \quad \phi,\psi \geq 0. \tag{2.41}$$

It was demonstrated in [48] that problem (2.41) can be expressed in more standard bound-constrained quadratic program as follows

$$\min_{w} \frac{1}{2} w^T Q w + h^T w, \quad \text{s.t} \quad w \geq 0, \tag{2.42}$$

$w = \begin{bmatrix} \phi \\ \psi \end{bmatrix}$, $h = \pi e_{2n} + \begin{pmatrix} -c \\ c \end{pmatrix}$, $c = M^T\vartheta$, $Q = \begin{bmatrix} M^TM & -M^TM \\ -M^TM & M^TM \end{bmatrix}$, where, the matrix Q is positive semi-definite. As a result, the problem in (2.42) is a convex quadratic programming, that is translated into the following problem of linear variable inequality (LVI) [119]. Find $z \in \mathbb{R}^n$, such that

$$(w'-w)^T(Qw+h) \geq 0 \quad \forall w' \geq 0. \tag{2.43}$$

In addition, problem in(2.42) is equivalent to the following linear complementary problem [119]. Find $w \in \mathbb{R}^n$,

$$w \geq 0, \quad Qw+h \geq 0, \quad \text{and} \quad w^T(Qw+h) = 0. \tag{2.44}$$

where, $w \in \mathbb{R}^n$ is the solution of problem in (2.44) if and only if it satisfies the nonlinear equations defined by

$$F(w) = \min\{w, Qw+h\} = 0, \tag{2.45}$$

where, F is a vector-valued function that is Lipschitz continuous and monotone as proved in Lemma 3 of [82] and Lemma 2.2 of [119] respectively. The "min" interpreted as component-wise minimum. Therefore, problem (2.39) can be translated into (1.3). Therefore, problem (2.39) can be solved by our proposed algorithms.

Many researchers developed methods for dealing with the ℓ_1-norm regularization problems in compressive sensing, which were successfully implemented.

Among them, Xiao and Zhu [120] proposed a CG method with ℓ_1-norm regularized CS problems. The conducted experiments showed that the presented method restored the disturbed signal better than the existing method proposed by Xiao et al. [119]. Consequently, Liu and Li in [73] modified the work of Xiao and Zhu [120] with the application in signal processing. Awwal et al. presented a method for compressive sensing sparse signal reconstruction in [11]. The method outperforms the method presented in [73] numerically. Aji et al. presented a method in [6] that restored the blurred image with higher quality than the existing method in [120]. Recently, Halilu et al. proposed a CG method in [65] that is more effective for restoring disturbed signals than the method method presented by Liu and Li in [73]. The reader may refer to [1, 12, 5, 118, 24] for further research and study on other ℓ_1-norm regularization problems in compressive sensing.

CHAPTER THREE

Signal Recovery with Convex Constrained Nonlinear Monotone Equations through Conjugate Gradient Hybrid Approach

3.1 Introduction

This chapter presents a CG method for solving monotone nonlinear equations with convex constraint (1.3). In recent years there is a vast application of conjugate gradient method to restore the disturbed signals in compressive sensing. This research aims at developing a scheme, which is more effective for restoring disturbed signals than the popular PCG method [73].To realize the desired goal, a new conjugate gradient approach combined with the projection scheme Solodov and Svaiter [97] for solving monotone nonlinear equations with convex constraints is presented. The main idea employed in this algorithm is to approximate the Jacobian matrix via acceleration parameter in order to propose an effective conjugate gradient parameter. In addition, the step length is calculated using inexact line search technique. The proposed approach is proved to be converged globally under some mild conditions . The numerical experiment showed the effectiveness of our method. Apart from generating search directions that are vital for global convergence, a significant contribution of the new method lies in its applications to solve the ℓ_1-norm regularization problem in signal recovery. Experiments with the scheme and the effective PCG solver, existing in literature, shows that the new method provides better results.

Several methods are used to solve (1.3) but the Newton and quasi-Newton methods are the widely used methods. These approaches are easy to implement and fast convergence from a reasonably good starting point. Nonetheless, at each iteration they require the computation of either the Jacobian or its approximation, so they're not ideal for solving large-scale problems. For this reason, conjugate gradient (CG) approaches have been proposed [120, 4, 35, 50, 88]. The methods are very promising to solve the large-scale problems because of their low storage requirements. Throughout this chapter, the space $\mathbb{R}^n$ denote the $n-$dimensional real space equipped with the Euclidean norm $\|\cdot\|$, $F_k = F(x_k)$, and $F'_k = F'(x_k)$.

Definition 3.1.1 *Let $\Omega \subset \mathbb{R}^n$ be a nonempty, closed and convex set. Then for any $x \in \mathbb{R}^n$, its projection onto Ω is given by*

$$P_\Omega(x) = arg\ min\{\|x-y\| : y \in \Omega\}. \tag{3.46}$$

The mapping $P_\Omega : \mathbb{R}^n \to \Omega$ is known as a projection operator which has nonexpansive property namely, for any $x, y \in \mathbb{R}^n$ it holds that

$$\|P_\Omega(x) - P_\Omega(y)\| \leq \|x-y\|, \tag{3.47}$$

and consequently, we have

$$\|P_\Omega(x) - y\| \leq \|x-y\|, \quad \forall y \in \Omega. \tag{3.48}$$

Assumption 3.1.2 *The following assumptions are needed.*

(A1) *The mapping F is Lipschitz continuous; namely, there exists a positive constant L such that*

$$\|F(x) - F(y)\| \leq L\|x-y\|, \quad \forall x, y \in \mathbb{R}^n. \tag{3.49}$$

(A2) *The mapping F is uniformly monotone, namely, there exists a positive constant c such that*

$$(x-y)^T(F(x)-F(y)) \geq c\|x-y\|^2, \quad \forall x,y \in \mathbb{R}^n. \tag{3.50}$$

(A3) *The solution set of (1.1) is nonempty and is denoted by S^x.*

(A4) *The mapping F has nonexpansive property; namely, for any $x,y \in \mathbb{R}^n$ it holds that*

$$\|F(x)-F(y)\| \leq \|x-y\|. \tag{3.51}$$

3.2 Derivation of the proposed method

Consider the derivative-free double direction method presented in [59]. The method developed a derivative-free method for solving systems of nonlinear equations via

$$F_k' \approx \gamma_k I, \tag{3.52}$$

where I is an identity matrix and $\gamma_k > 0$ is an acceleration parameter. The method in [59] produces a sequence of iterates $\{x_k\}$ such that $x_{k+1} = x_k + (\alpha_k + \alpha_k^2\gamma_k)d_k$ and the direction d_k is given as

$$d_k = -\gamma_k^{-1}F_k, \tag{3.53}$$

where γ_k is given by

$$\gamma_k = \frac{y_{k-1}^T y_{k-1}}{y_{k-1}^T s_{k-1}}, \tag{3.54}$$

where, $y_{k-1} = F_k - F_{k-1}$ and $s_{k-1} = x_k - x_{k-1}$.

Despite the fact that the method [59] has strong convergence properties, its numerical performance is weak when α_k approaches to zero. For this reason, we are motivated to propose a hybrid conjugate gradient method with strong convergence properties and effective numerical results.

To present a hybrid form of the method in [59], the mapping T is assumed to be defined by an improved double direction method as $T(v_k) = v_k - (\alpha_k +$

$\alpha_k^2\gamma_k)\gamma_k^{-1}F_k$. By this assumption and (2.18) we have

$$\begin{cases} x_1 = x \in \Omega, \\ v_k = (1-\eta_k)x_k + \eta_k T x_k = x_k - \eta_k(\alpha_k + \alpha_k^2\gamma_k)\gamma_k^{-1}F_k, \\ x_{k+1} = T(v_k) = v_k - (\alpha_k + \alpha_k^2\gamma_k)\gamma_k^{-1}F_k, \quad k \in \mathbb{N}. \end{cases} \tag{3.55}$$

From the second and third equations in (6.191) we obtain the iterative scheme,

$$x_{k+1} = x_k - t_k(\alpha_k + \alpha_k^2\gamma_k)\gamma_k^{-1}F_k, \tag{3.56}$$

where, $t_k = (\eta_k + 1) \in (1,2)$ is a correction parameter. we can easily show that, the search direction in (6.192) is defined as:

$$d_k = -t_k\gamma_k^{-1}F_k. \tag{3.57}$$

A hybrid conjugate gradient method is now presented. This is made possible by combining (1.12) and (3.57) as follows:

$$t_k\gamma_k^{-1}F_k = (F_k - \beta_k d_{k-1}), \tag{3.58}$$

by substituting (3.54) into (3.58) we have

$$t_k(y_{k-1}^T s_{k-1})F_k = y_{k-1}^T y_{k-1}(F_k - \beta_k d_{k-1}). \tag{3.59}$$

Observe that after multiplying (3.59) by y_{k-1}^T it can be written as

$$\beta_k(y_{k-1}^T y_{k-1})(y_{k-1}^T d_{k-1}) = (y_{k-1}^T y_{k-1})(y_{k-1}^T F_k) - t_k(y_{k-1}^T F_k)(y_{k-1}^T s_{k-1}). \tag{3.60}$$

Applying some algebraic calculation, we can write the proposed CG parameter and search direction as

$$\beta_k = \frac{(\|y_{k-1}\|^2 - t_k y_{k-1}^T s_{k-1})y_{k-1}^T F_k}{\|y_{k-1}\|^2 y_{k-1}^T d_{k-1}}, \tag{3.61}$$

$$d_k = \begin{cases} -F_k, & \text{if} \quad k = 0, \\ -\theta_k F_k + \dfrac{(\|y_{k-1}\|^2 - t_k y_{k-1}^T s_{k-1}) y_{k-1}^T F_k}{\|y_{k-1}\|^2 y_{k-1}^T d_{k-1}} d_{k-1}, & \text{if} \quad k \geq 1. \end{cases} \tag{3.62}$$

where $\theta_k = \frac{s_{k-1}^T s_{k-1}}{y_{k-1}^T s_{k-1}}$. (see [111]).

Next, the algorithm of the proposed method is specified as follows:

Algorithm 1: Conjugate gradient hybrid method(CHCG)

Input: Given $x_0 \in R^n$, $d_0 = -F_0$, $\gamma_0 = 0.01$, $t = t_k \in (1,2)$, $\rho \in (0,1)$, $\sigma > 0$, $\varepsilon = 10^{-10}$, $\xi > 0$, set $k = 0$.

Step 1: Compute F_k. If $\|F_k\| \leq \varepsilon$ then stop, else goto the next step.

Step 2: Let $\mu_k = (\alpha_k + \alpha_k^2 \gamma_k)$ and set $z_k = x_k + \mu_k d_k$, where, $\alpha_k = \xi \rho^{m_k}$ with m_k being the smallest nonnegative integer m satisfying

$$-F(x_k + \mu_k d_k)^T d_k \geq \sigma \mu_k \|d_k\|^2. \tag{3.63}$$

Step 3: If $z_k \in \Omega$ and $\|F(z_k)\| \leq \varepsilon$, then stop, otherwise go to the next Step.

Step 4: Compute the next iterate by

$$x_{k+1} = P_\Omega[x_k - \lambda_k F(z_k)], \tag{3.64}$$

where $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step 5: Compute d_{k+1} using (3.62).

Step 6: Determine $\gamma_{k+1} = \dfrac{y_k^T y_k}{y_k^T s_k}$, where, $s_k = z_k - x_k$ and $y_k = (F(z_k) - F(x_k))$.

Step 7: Set k=k+1, and go to STEP 2.

Remark 3.2.1 *We give the following remarks*

(a) From (3.50), we have

$$y_{k-1}^T s_{k-1} = s_{k-1}^T (F(z_{k-1}) - F(x_{k-1})) \geq c\|s_{k-1}\|^2 > 0. \tag{3.65}$$

From (3.51) and (3.65) we have

$$y_{k-1}^T s_{k-1} = (F(z_{k-1}) - F(x_{k-1}))^T (z_{k-1} - x_{k-1}) \leq \|s_{k-1}\|^2. \tag{3.66}$$

Also, from (3.65) and (3.66) it is simple to check that, $\|s_{k-1}\|^2 \geq y_{k-1}^T s_{k-1} \geq c\|s_{k-1}\|^2$*, this implies that* $\frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \leq \frac{1}{c}$ *and* $\frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \geq 1$*. This means* θ_k *in (3.62) is well defined. Therefore, we have*

$$1 \leq \theta_k \leq \frac{1}{c}. \tag{3.67}$$

(b) From assumption (A4) we have

$$\|y_{k-1}\| = \|F(z_{k-1}) - F(x_{k-1})\| \leq \|s_{k-1}\| = \mu_{k-1}\|d_{k-1}\|. \tag{3.68}$$

(c) From assumption (A2) we have

$$y_{k-1}^T d_{k-1} = (F(z_{k-1}) - F(x_{k-1}))^T \frac{s_{k-1}}{\mu_{k-1}} \geq \frac{c\|s_{k-1}\|^2}{\mu_{k-1}} = c\mu_{k-1}\|d_{k-1}\|^2 > 0. \tag{3.69}$$

Since $y_{k-1}^T d_{k-1} > 0$*, the proposed* β_k *in (3.61) is well defined.*

(d) From assumption (A3) and (A4) we have

$$\frac{\|y_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \leq \frac{\|s_{k-1}\|^2}{c\|s_{k-1}\|^2} = \frac{1}{c}. \tag{3.70}$$

Therefore, from (3.54) and (3.70) we have

$$\gamma_k = \frac{\|y_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \leq \frac{1}{c}. \tag{3.71}$$

3.3 Convergence Analysis of Algorithm 1 (CHCG)

In this subsection, the global convergence of the CHCG Algorithm is established.

Lemma 3.3.1 *Suppose assumptions (A2)-(A4) hold and $\{x_k\}$ be generated by CHCG Algorithm, then the search direction d_k fulfills the following property*

$$F_k^T d_k < 0. \tag{3.72}$$

Proof . Here two cases are to be considered:
Case 1: For $k = 0$, by (3.62) we have $F_0^T d_0 = -\|F_0\|^2 < 0$.
Case 2: From (3.71) we have $\|y_{k-1}\|^2 \leq \frac{1}{c} y_{k-1}^T s_{k-1}$.
Now, from (3.62) and since $t = t_k \in (1,2)$, we have

$$\begin{aligned}
F_k^T d_k &= -\theta_k \|F_k\|^2 + \frac{(\|y_{k-1}\|^2 - t y_{k-1}^T s_{k-1})(y_{k-1}^T F_k)(F_k^T d_{k-1})}{\|y_{k-1}\|^2 (y_{k-1}^T d_{k-1})}, \\
&\leq -\theta_k \|F_k\|^2 + \frac{(\frac{1}{c} y_{k-1}^T s_{k-1} - t y_{k-1}^T s_{k-1})(y_{k-1}^T F_k)(F_k^T d_{k-1})}{\|y_{k-1}\|^2 (y_{k-1}^T d_{k-1})}, \\
&\leq -\theta_k \|F_k\|^2 + \frac{(\frac{1}{c} y_{k-1}^T s_{k-1} - \frac{1}{c} y_{k-1}^T s_{k-1})(y_{k-1}^T F_k)(F_k^T d_{k-1})}{\|y_{k-1}\|^2 (y_{k-1}^T d_{k-1})}, \\
&\leq -\theta_k \|F_k\|^2, \\
&\leq -\|F_k\|^2.
\end{aligned} \tag{3.73}$$

The last inequality follows from (3.67). ∎

Using the next lemma, we demonstrate that the line search used in CHCG Algorithm is well-defined .

Lemma 3.3.2 *Suppose assumptions (A2)-(A4) hold. Then there exists a step length α_k satisfying (3.63) for all $k \geq 0$.*

Proof Going by contradiction, we assume that there exists a constant $k_0 \geq 0$, such that given any nonnegative integer m, we have

$$-F(x_{k_0} + (\xi\rho^m + (\xi\rho^m)^2 \gamma_{k_0}) d_{k_0})^T d_{k_0} < \sigma(\xi\rho^m + (\xi\rho^m)^2 \gamma_{k_0}) \|d_{k_0}\|^2. \tag{3.74}$$

Since $\rho \in (0,1)$, by using assumption $(A2)$, (3.63) and letting $m \to \infty$, we get

$$-F(x_{k_0})^T d_{k_0} \leq 0. \tag{3.75}$$

Also from (6.203), it follows that

$$-F(x_{k_0})^T d_{k_0} \geq \|F(x_{k_0})\|^2 > 0, \tag{3.76}$$

which clearly contradicts (3.75). Hence, the line search is well-defined. ■

Lemma 3.3.3 *Suppose assumptions (A2)-(A4) hold and let sequences* $\{x_k\}$ *and* $\{z_k\}$ *be generated by CHCG Algorithm, then* $\{x_k\}$ *and* $\{z_k\}$ *are bounded. Moreover, we have*

$$\lim_{k\to\infty} \|x_k - z_k\| = 0, \tag{3.77}$$

and

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{3.78}$$

Proof First, we show the boundedness of the sequences $\{x_k\}$ and $\{z_k\}$. Let $\bar{x} \in S^x$ be any solution of (1.3). Then by monotonicity of F we can write

$$(x_k - \bar{x})^T F(z_k) \geq (x_k - z_k)^T F(z_k). \tag{3.79}$$

Using the line search condition (3.63) and definition of z_k, we have

$$(x_k - z_k)^T F(z_k) \geq \sigma \mu_k^2 \|d_k\|^2 > 0. \tag{3.80}$$

Also, using (3.48) and(3.64) we have

$$\begin{aligned}
\|x_{k+1} - \bar{x}\|^2 &= \|P_\Omega(x_k - \lambda_k F(z_k)) - \bar{x}\|^2, \\
&\leq \|x_k - \lambda_k F(z_k) - \bar{x}\|^2, \\
&= \|x_k - \bar{x}\|^2 - 2\lambda_k (x_k - \bar{x})^T F(z_k) + \lambda_k^2 \|F(z_k)\|^2, \\
&\leq \|x_k - \bar{x}\|^2 - 2\lambda_k (x_k - z_k)^T F(r_z) + \lambda_k^2 \|F(z_k)\|^2, \\
&= \|x_k - \bar{x}\|^2 - \frac{((x_k - z_k)^T F(z_k))^2}{\|F(z_k)\|^2}, \\
&\leq \|x_k - \bar{x}\|^2 - \sigma^2 \|x_k - z_k\|^4,
\end{aligned} \tag{3.81}$$

for which we obtain

$$\|x_{k+1} - \bar{x}\| \leq \|x_k - \bar{x}\|. \tag{3.82}$$

This recursively implies that $\|x_k - \bar{x}\| \leq \|x_0 - \bar{x}\| \; \forall k$.

So, the sequence $\{\|x_k - \bar{x}\|\}$ is clearly a decreasing, which implies that $\{x_k\}$ is bounded. Furthermore, utilizing assumption (A3), (A4) and (3.82) we have

$$\|F(x_k)\| = \|F(x_k) - F(\bar{x})\| \leq \|x_k - \bar{x}\| \leq \|x_0 - \bar{x}\|. \tag{3.83}$$

Let $m_1 = \|x_0 - \bar{x}\|$, then we have

$$\|F(x_k)\| \leq m_1. \tag{3.84}$$

Moreover, from the definition of z_k, monotonicity of F, (3.80) and the Cauchy-Schwarz inequality, we have

$$\sigma\|x_k - z_k\| = \frac{\sigma\|\mu_k d_k\|^2}{\|x_k - z_k\|} \leq \frac{(x_k - z_k)^T F(z_k)}{\|x_k - z_k\|} \leq \frac{(x_k - z_k)^T F(x_k)}{\|x_k - z_k\|} \leq \|F(x_k)\|. \tag{3.85}$$

So, using the boundedness of the sequences $\{x_k\}$,(6.207) and (3.85), one see that $\{z_k\}$ is bounded. Therefore, by the boundedness of the $\{z_k\}$, the sequence $\{\|z_k - \bar{x}\|\}$ is also bounded, i.e., there exists a constant $\kappa > 0$ for any $\bar{x} \in \Omega$, such that

$$\|z_k - \bar{x}\| \leq \kappa. \tag{3.86}$$

From (3.49) and (3.86), we have

$$\|F(z_k)\| = \|F(z_k) - F(\bar{x})\| \leq \|z_k - \bar{x}\| \leq \kappa. \tag{3.87}$$

Also, from (3.81), (3.85), (3.87), we obtain

$$\frac{\sigma^2}{(\kappa)^2} \sum_{k=0}^{\infty} \|x_k - z_k\|^4 \leq \sum_{k=0}^{\infty} \frac{((x_k - z_k)^T F(z_k))^2}{\|F(z_k)\|^2} \leq \sum_{k=0}^{\infty} (\|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2) < \infty. \tag{3.88}$$

This implies (3.77) .
On the other hand, from (3.48), the definition of λ_k we have

$$\begin{aligned} \|x_{k+1} - x_k\| &= \|P_\Omega(x_k - \lambda_k F(z_k)) - x_k\|, \\ &\leq \|x_k - \lambda_k F(z_k) - x_k\|, \\ &= \|\lambda_k F(z_k)\|, \\ &\leq \|x_k - z_k\|. \end{aligned} \tag{3.89}$$

This implies (3.78) .
Notwithstanding, from (3.77) and the definition of z_k we have

$$\lim_{k\to\infty} \mu_k \|d_k\| = 0. \tag{3.90}$$

■

Lemma 3.3.4 *Suppose assumption (A2)-(A4) hold and $\{x_k\}$ be generated by CHCG Algorithm. Then there exists some positive constants M such that for all $k > 0$,*

$$\|d_k\| \leq M. \tag{3.91}$$

Proof Here two cases are to be considered:
Case 1: For $k = 0$, by (3.62) and (3.84) we have $\|d_0\| = -\|F_0\| < m_1$.

Case 2: For $k \geq 1$, also from (3.62),(3.84) and remarks (a)-(d) we have

$$\begin{aligned}
\|d_k\| &= \left\| -\theta_k F_k + \frac{(\|y_{k-1}\|^2 - t y_{k-1}^T s_{k-1}) y_{k-1}^T F_k}{\|y_{k-1}\|^2 (y_{k-1}^T d_{k-1})} d_{k-1} \right\|, \\
&\leq \theta_k \|F_k\| + \left| \frac{y_{k-1}^T F_k}{y_{k-1}^T d_{k-1}} - \frac{t(y_{k-1}^T s_{k-1})(y_{k-1}^T F_k)}{\|y_{k-1}\|^2 (y_{k-1}^T d_{k-1})} \right| \|d_{k-1}\|, \\
&\leq \theta_k \|F_k\| + \left[\frac{|y_{k-1}^T F_k|}{c\mu_{k-1}\|d_{k-1}\|^2} + \frac{t|y_{k-1}^T s_{k-1}||y_{k-1}^T F_k|}{\|y_{k-1}\|^2 (c\mu_{k-1}\|d_{k-1}\|^2)} \right] \|d_{k-1}\|, \\
&\leq \theta_k \|F_k\| + \left[\frac{\|y_{k-1}\|\|F_k\|}{c\mu_{k-1}\|d_{k-1}\|^2} + \frac{t\|y_{k-1}\|^2 \|s_{k-1}\|\|F_k\|}{\|y_{k-1}\|^2 (c\mu_{k-1}\|d_{k-1}\|^2)} \right] \|d_{k-1}\|, \\
&\leq \theta_k \|F_k\| + \left[\frac{(\mu_{k-1}\|d_{k-1}\|)\|F_k\|}{c\mu_{k-1}\|d_{k-1}\|^2} + \frac{t(\mu_{k-1}\|d_{k-1}\|)\|F_k\|}{c\mu_{k-1}\|d_{k-1}\|^2} \right] \|d_{k-1}\|, \\
&\leq \left[\theta_k + \frac{(1+t)}{c} \right] \|F_k\|, \\
&\leq \left[\frac{(2+t)}{c} \right] m_1.
\end{aligned} \tag{3.92}$$

The last inequality follows from (3.67). Taking $M = \left[\frac{(2+t)}{c} \right] m_1$, we have (3.91). ■

Theorem 3.3.5 *Suppose assumption (A2)-(A4) hold and $\{x_k\}$ be generated by Algorithm 7. Then*

$$\liminf_{k \to \infty} \|F_k\| = 0. \tag{3.93}$$

Proof Going by contradiction, suppose that (3.93) is not true, then there exists $n_0 > 0$ such that

$$\|F_k\| \geq \delta_0 \quad \text{holds}, \quad \forall k > 0. \tag{3.94}$$

By (3.73) and Cauchy-Schwartz inequality, we have

$$\|F_k\|^2 \leq -F_k^T d_k \leq \|F_k\|\|d_k\|, \quad \forall k \in \mathbf{N}. \tag{3.95}$$

So,

$$\|d_k\| \geq \delta_0 > 0. \tag{3.96}$$

If $\mu_k \neq \xi$, then by the definition of $\mu_k, \rho^{-1}\mu_k$ does not satisfies (3.63) i.e.,

$$-F(x_k + \rho^{-1}\mu_k d_k)^T d_k < \sigma\rho^{-1}\mu_k \|d_k\|^2.$$

Now, combining with (3.95), we have

$$\begin{aligned}\|F_k\|^2 &= -F_k^T d_k, \\ &= (F(x_k + \rho^{-1}\mu_k d_k) - F_k)^T d_k - F(x_k + \rho^{-1}\mu_k d_k)^T d_k, \\ &\leq \rho^{-1}\mu_k \|d_k\|^2 + \sigma\rho^{-1}\mu_k \|d_k\|^2.\end{aligned}$$

The above inequality implies

$$\mu_k \geq \frac{\rho}{(1+\sigma)} \frac{\|F_k\|^2}{\|d_k\|^2}.$$

From (3.91) and (3.94) we have,

$$\mu_k \|d_k\| \geq \frac{\rho}{(1+\sigma)} \frac{\|F_k\|^2}{\|d_k\|} \geq \frac{\rho}{(1+\sigma)} \frac{\delta_0^2}{M}.$$

This contradicts (3.90). Therefore (3.93) holds.
The proof is completed. ■

3.4 Numerical Experiments

Some numerical results are provided in this subsection to show the effectiveness of the proposed method by comparing it with the following existing methods in the literature.

- A projection method for convex constrained monotone nonlinear equations with applications (**PCG**) [73].
- An efficient three-term conjugate gradient method for nonlinear monotone equations with convex constraints (**ETTCG**) [52].

The computer codes used are written in Matlab 9.4.0 (R2018a) and run on a personal computer equipped with a 1.80 GHz CPU processor and 8 GB RAM. The

same line search is used (3.63), when implementing the three algorithms in this experiments, and the following parameters are set $\xi = 0.5$, $\sigma = 10^{-4}$ and $\rho = 0.9$, we however set $t = 1.2$ in our algorithm. The iteration is set to stop for all the methods if $\|F_k\| \leq 10^{-10}$ or when the iterations exceed 1000. We use the symbol '-' to represents failure due to (i) Memory requirement (ii) Number of iterations exceed 1000. We have tried the three methods on the previous four test problems with different initial guess and dimension (n values). The test problems 1,2 and 3 below are from [3] while the remaining one is from [29]. To show the extensive numerical experiments of CHCG, PCG and ETTCG methods, the experimentation was carried out with the dimensions 1000, 5000, 10,000, 50,000 and 100,000. The reported results of the three (3) methods are shown in Tables 3.2-3.5. From the Tables, "IT" represents the total number of iterations, "TIME(s)" represents the CPU time (in seconds), "IP" represents the initial points and "FV" represents the functions evaluations, and $\|F_k\|$ represents the norm of function at the termination point. We use the following initial points for the test problems:

Table 3.1: Starting points used to test problems

Initial Points (IP)	Values
$x1$	$(0.5, 0.5, ..., 0.5)^T$
$x2$	$(0.2, 0.2, ..., 0.2)^T$
$x3$	$(1, 1, ..., 1)^T$
$x4$	$\left(\frac{2}{5}, \frac{2}{5}, ..., \frac{2}{5}\right)^T$
$x5$	$\left(0, \frac{1}{2}, \frac{2}{3}, ..., 1\right)^T$
$x6$	$\left(\frac{1}{4}, \frac{-1}{4}, ..., \frac{(-1)^n}{4}\right)^T$
$x7$	$(4, 4, ..., 4)^T$

It should be noted here, that the parameters employed for the experiments conducted are carefully selected to yield the most desired results. In addition, the initial starting points used were selected in an evenly spaced as well as mixed valued fashion to avoid a biased computation.

The following test problems were used in the experiments:

Problem 1

$F_1(x) = e^{x_1} - 1,$

$F_i(x) = e^{x_i} + x_{i-1} - 1, \quad i = 2, \ldots, n-1,$

where $\Omega = \mathbb{R}^n_+$.

Problem 2

$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1, \ldots, n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 3

$F_i(x) = e^{x_i} - 1, \quad i = 1, 2, \ldots, n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 4

$F_i(x) = \min(\min(|x_i|, x_i^2), \max(|x_i|, x_i^3)), \quad i = 1, 2 \ldots, n,$

where $\Omega = \mathbb{R}^n_-$.

Table 3.2: Numerical results of CHCG, PCG and ETTCG methods for problem 1

		CHCG				PCG				ETTCG			
Dimension	IP	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$
1000	x1	2	20	0.163684	0	9	13	0.118308	7.16E-12	11	26	0.051611	2.3E-11
	x2	2	19	0.017372	0	8	11	0.022756	6.17E-11	-	-	-	-
	x3	2	21	0.037511	0	9	15	0.01738	5.17E-11	-	-	-	-
	x4	2	20	0.010232	0	9	13	0.016886	7.36E-12	-	-	-	-
	x4	2	21	0.024716	0	21	38	0.039699	1.38E-11	13	29	0.029597	5.43E-11
	x6	1	7	0.010254	0	35	37	0.036661	7.32E-11	32	34	0.034886	7.22E-11
	x7	5	61	0.01853	0	8	33	0.015206	5.74E-12	2	25	0.012648	0
5000	x1	2	20	0.02189	0	9	13	0.029987	1.46E-11	8	19	0.036214	0
	x2	2	19	0.017597	0	9	12	0.025697	5.97E-12	7	16	0.033642	0
	x3	2	21	0.01811	0	9	14	0.03354	1.22E-11	-	-	-	-
	x4	2	20	0.016753	0	9	13	0.026176	1.58E-11	9	21	0.043088	0
	x4	2	21	0.017572	0	22	29	0.061706	1.5E-11	13	29	0.053321	1.2E-11
	x6	1	7	0.008953	0	35	37	0.091173	7.09E-11	32	34	0.093882	7.16E-11
	x7	5	61	0.04442	0	7	32	0.034183	4.68E-11	2	25	0.034272	0
10000	x1	2	20	0.027082	0	9	13	0.051895	2.04E-11	-	-	-	-
	x2	2	19	0.029402	0	9	12	0.043989	8.13E-12	7	16	0.05215	0
	x3	2	21	0.028343	0	9	14	0.0453	2.29E-11	-	-	-	-
	x4	2	20	0.027543	0	9	13	0.047154	2.23E-11	8	19	0.061535	0
	x4	2	21	0.030014	0	21	32	0.10264	5.12E-11	12	29	0.091707	1.1E-11
	x6	1	7	0.013536	0	35	37	0.14576	7.06E-11	32	34	0.15223	7.15E-11
	x7	5	61	0.071851	0	7	32	0.058053	3.31E-11	2	25	0.051753	0
50000	x1	2	20	0.108286	0	9	13	0.16896	4.52E-11	8	19	0.232204	0
	x2	2	19	0.094496	0	9	12	0.158822	1.76E-11	7	16	0.196321	0
	x3	2	21	0.103246	0	9	14	0.163749	2.54E-11	-	-	-	-
	x4	2	20	0.100786	0	9	13	0.16446	4.97E-11	9	21	0.260351	0
	x4	2	21	0.101266	0	26	31	0.429475	8.05E-12	12	27	0.321023	9.38E-11
	x6	1	7	0.042233	0	35	37	0.537817	7.03E-11	32	34	0.56907	7.14E-11
	x7	5	61	0.274494	0	7	32	0.209848	1.48E-11	2	25	0.208295	0
100000	x1	2	20	0.189274	0	9	13	0.322127	6.39E-11	-	-	-	-
	x2	2	19	0.195335	0	9	12	0.32179	2.47E-11	-	-	-	-
	x3	2	21	0.186434	0	9	14	0.338365	2.72E-11	8	20	0.444494	0
	x4	2	20	0.183533	0	9	13	0.31985	7.01E-11	8	19	0.495861	0
	x4	2	21	0.203235	0	27	32	0.861223	6.6E-12	12	27	0.627386	9.67E-11
	x6	1	7	0.077777	0	35	37	1.052519	7.03E-11	32	34	1.118661	7.14E-11
	x7	5	61	0.509883	0	7	32	0.424138	1.05E-11	2	25	0.389371	0

Table 3.3: Numerical results of CHCG, PCG and ETTCG methods for problem 2

		CHCG				PCG				ETTCG			
Dimension	IP	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$
1000	x1	2	15	0.029374	0	40	42	0.036352	7.66E-11	37	39	0.039826	5.45E-11
	x2	2	15	0.00737	0	39	41	0.048604	6.03E-11	35	37	0.033532	9.12E-11
	x3	2	15	0.007552	0	41	43	0.036147	6.96E-11	37	39	0.036506	9.31E-11
	x4	2	15	0.006319	0	40	42	0.033371	6.25E-11	36	38	0.035451	8.88E-11
	x4	2	15	0.008652	0	41	43	0.036249	6.93E-11	37	39	0.035895	9.27E-11
	x6	1	10	0.003742	0	1	6	0.004953	0	1	6	0.004605	0
	x7	2	20	0.006947	0	38	41	0.037409	7.74E-11	35	38	0.036112	6.3E-11
5000	x1	2	15	0.017194	0	41	43	0.09602	9.1E-11	38	40	0.107244	6.09E-11
	x2	2	15	0.017678	0	40	42	0.094852	7.17E-11	37	39	0.097091	5.1E-11
	x3	2	15	0.016511	0	42	44	0.120174	8.27E-11	39	41	0.102489	5.2E-11
	x4	2	15	0.014199	0	41	43	0.100866	7.42E-11	37	39	0.104148	9.93E-11
	x4	2	15	0.018053	0	42	44	0.119664	8.26E-11	39	41	0.105474	5.2E-11
	x6	1	10	0.008182	0	1	6	0.012946	0	1	6	0.008185	0
	x7	2	20	0.029637	0	39	42	0.088982	9.2E-11	36	39	0.112181	7.04E-11
10000	x1	2	15	0.027072	0	42	44	0.174262	6.83E-11	38	40	0.201885	8.61E-11
	x2	2	15	0.026655	0	41	43	0.170038	5.38E-11	37	39	0.196989	7.21E-11
	x3	2	15	0.027178	0	43	45	0.17995	6.21E-11	39	41	0.208645	7.36E-11
	x4	2	15	0.025912	0	42	44	0.173002	5.57E-11	38	40	0.189378	7.02E-11
	x4	2	15	0.026861	0	43	45	0.203715	6.21E-11	39	41	0.200187	7.36E-11
	x6	1	10	0.01539	0	1	6	0.010418	0	1	6	0.013979	0
	x7	2	20	0.057997	0	40	43	0.170794	6.91E-11	36	39	0.180222	9.96E-11
50000	x1	2	15	0.099602	0	43	45	0.711364	8.12E-11	39	41	0.746359	9.63E-11
	x2	2	15	0.096692	0	42	44	0.66636	6.39E-11	38	40	0.721019	8.06E-11
	x3	2	15	0.098632	0	44	46	0.75902	7.38E-11	40	42	0.756109	8.23E-11
	x4	2	15	0.098672	0	43	45	0.691368	6.62E-11	39	41	0.730112	7.85E-11
	x4	2	15	0.103324	0	44	46	0.696946	7.38E-11	40	42	0.755147	8.23E-11
	x6	1	10	0.054669	0	1	6	0.038919	0	1	6	0.049837	0
	x7	2	20	0.240239	0	41	44	0.649783	8.21E-11	38	41	0.73583	5.57E-11
100000	x1	2	15	0.184932	0	44	46	1.317218	6.1E-11	40	42	1.420589	6.81E-11
	x2	2	15	0.177485	0	42	44	1.285065	9.04E-11	39	41	1.390621	5.7E-11
	x3	2	15	0.194041	0	45	47	1.362545	5.54E-11	41	43	1.441766	5.82E-11
	x4	2	15	0.189785	0	43	45	1.303302	9.36E-11	40	42	1.425622	5.55E-11
	x4	2	15	0.189841	0	45	47	1.374635	5.54E-11	41	43	1.438479	5.82E-11
	x6	1	10	0.09437	0	1	6	0.07481	0	1	6	0.097027	0
	x7	2	20	0.476423	0	42	45	1.332267	6.17E-11	38	41	1.351728	7.87E-11

Table 3.4: Numerical results of CHCG, PCG and ETTCG methods for problem 3

		CHCG				PCG				ETTCG			
Dimension	IP	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$
1000	x1	2	16	0.021986	0	39	41	0.036379	9.02E-11	36	38	0.035475	6.77E-11
	x2	2	15	0.006916	0	38	40	0.036641	9.39E-11	35	37	0.030722	7.51E-11
	x3	2	18	0.005059	0	39	41	0.033073	7.45E-11	36	38	0.036125	5.62E-11
	x4	2	16	0.006708	0	39	41	0.033011	8.08E-11	36	38	0.035223	6.07E-11
	x4	2	18	0.006346	0	39	41	0.033247	7.83E-11	36	38	0.035404	5.92E-11
	x6	1	2	0.002984	0	1	2	0.002689	0	1	2	0.002352	0
	x7	1	13	0.005176	0	1	13	0.00729	0	1	13	0.005587	0
5000	x1	2	16	0.014442	0	41	43	0.092703	5.69E-11	37	39	0.083218	7.56E-11
	x2	2	15	0.013644	0	40	42	0.090625	5.93E-11	36	38	0.085523	8.4E-11
	x3	2	18	0.013496	0	40	42	0.079467	8.85E-11	37	39	0.088383	6.29E-11
	x4	2	16	0.019168	0	40	42	0.084485	9.6E-11	37	39	0.091678	6.78E-11
	x4	2	18	0.014326	0	40	42	0.085871	8.96E-11	37	39	0.10509	6.37E-11
	x6	1	2	0.004804	0	1	2	0.005276	0	1	2	0.004213	0
	x7	1	13	0.010387	0	1	13	0.009144	0	1	13	0.014674	0
10000	x1	2	16	0.021618	0	41	43	0.144399	8.05E-11	38	40	0.157972	5.35E-11
	x2	2	15	0.020327	0	40	42	0.137939	8.38E-11	37	39	0.148613	5.94E-11
	x3	2	18	0.019475	0	41	43	0.143064	6.65E-11	37	39	0.14805	8.89E-11
	x4	2	16	0.023093	0	41	43	0.147929	7.21E-11	37	39	0.148313	9.59E-11
	x4	2	18	0.021758	0	41	43	0.141932	6.69E-11	37	39	0.147792	8.95E-11
	x6	1	2	0.008368	0	1	2	0.006086	0	1	2	0.006806	0
	x7	1	13	0.015422	0	1	13	0.018181	0	1	13	0.024177	0
50000	x1	2	16	0.065526	0	42	44	0.600517	9.56E-11	39	41	0.590405	5.98E-11
	x2	2	15	0.075864	0	41	43	0.58302	9.95E-11	38	40	0.571569	6.64E-11
	x3	2	18	0.072682	0	42	44	0.537692	7.89E-11	38	40	0.557761	9.94E-11
	x4	2	16	0.066479	0	42	44	0.531169	8.56E-11	39	41	0.581891	5.36E-11
	x4	2	18	0.07032	0	42	44	0.543927	7.91E-11	38	40	0.558955	9.95E-11
	x6	1	2	0.016532	0	1	2	0.019537	0	1	2	0.019875	0
	x7	1	13	0.060173	0	1	13	0.058134	0	1	13	0.095164	0
100000	x1	2	16	0.120074	0	43	45	1.049815	7.18E-11	39	41	1.166702	8.45E-11
	x2	2	15	0.1372	0	42	44	1.032128	7.48E-11	38	40	1.185053	9.39E-11
	x3	2	18	0.134959	0	43	45	1.058502	5.93E-11	39	41	1.116671	7.03E-11
	x4	2	16	0.124429	0	43	45	1.057841	6.44E-11	39	41	1.114056	7.58E-11
	x4	2	18	0.133714	0	43	45	1.04834	5.94E-11	39	41	1.109764	7.03E-11
	x6	1	2	0.038047	0	1	2	0.038581	0	1	2	0.040341	0
	x7	1	13	0.10898	0	1	13	0.106661	0	1	13	0.179577	0

Table 3.5: Numerical results of CHCG, PCG and ETTCG methods for problem 4

		CHCG				PCG				ETTCG			
Dimension	IP	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$	ITER	FVAL	TIME	$\|F_k\|$
1000	x1	1	2	0.019786	0	1	2	0.005798	0	1	2	0.004697	0
	x2	1	2	0.00311	0	1	2	0.00499	0	1	2	0.00424	0
	x3	1	3	0.003708	0	1	2	0.003718	0	1	2	0.003332	0
	x4	1	2	0.0039	0	1	2	0.003028	0	1	2	0.004362	0
	x4	6	51	0.075968	0	1	2	0.006383	0	1	2	0.004437	0
	x6	2	15	0.014309	0	39	41	0.07612	7.61E-11	36	38	0.075619	5.75E-11
	x7	1	3	0.003759	0	1	2	0.003951	0	1	2	0.003512	0
	x8	1	3	0.004116	0	1	2	0.003798	0	1	2	0.003283	0
5000	x1	1	2	0.009895	0	1	2	0.009894	0	1	2	0.00961	0
	x2	1	2	0.00818	0	1	2	0.010049	0	1	2	0.010358	0
	x3	1	3	0.008176	0	1	2	0.009523	0	1	2	0.008871	0
	x4	1	2	0.008361	0	1	2	0.009188	0	1	2	0.009654	0
	x4	6	52	0.10194	0	1	2	0.009636	0	1	2	0.008926	0
	x6	2	15	0.033915	0	40	42	0.280228	9.04E-11	37	39	0.280049	6.43E-11
	x7	1	3	0.008834	0	1	2	0.00833	0	1	2	0.008089	0
	x8	1	3	0.011054	0	1	2	0.009687	0	1	2	0.009188	0
10000	x1	1	2	0.015593	0	1	2	0.013689	0	1	2	0.015973	0
	x2	1	2	0.014588	0	1	2	0.015749	0	1	2	0.017866	0
	x3	1	3	0.014447	0	1	2	0.010881	0	1	2	0.014791	0
	x4	1	2	0.013478	0	1	2	0.016782	0	1	2	0.017438	0
	x4	6	52	0.185686	0	1	2	0.014733	0	1	2	0.015988	0
	x6	2	15	0.057634	0	41	43	0.480515	6.79E-11	37	39	0.514008	9.09E-11
	x7	1	3	0.012115	0	1	2	0.012953	0	1	2	0.014184	0
	x8	1	3	0.017172	0	1	2	0.014655	0	1	2	0.016205	0
50000	x1	1	2	0.05701	0	1	2	0.053925	0	1	2	0.068732	0
	x2	1	2	0.054249	0	1	2	0.055785	0	1	2	0.066109	0
	x3	1	3	0.050533	0	1	2	0.043045	0	1	2	0.05295	0
	x4	1	2	0.056469	0	1	2	0.057981	0	1	2	0.064508	0
	x4	5	51	0.739303	0	1	2	0.056216	0	1	2	0.062733	0
	x6	2	15	0.245875	0	42	44	2.118536	8.06E-11	39	41	2.375988	5.08E-11
	x7	1	3	0.047474	0	1	2	0.047494	0	1	2	0.052935	0
	x8	1	3	0.060215	0	1	2	0.05784	0	1	2	0.07239	0
100000	x1	1	2	0.113308	0	1	2	0.113189	0	1	2	0.129614	0
	x2	1	2	0.106407	0	1	2	0.111957	0	1	2	0.132337	0
	x3	1	3	0.094876	0	1	2	0.081419	0	1	2	0.102163	0
	x4	1	2	0.103429	0	1	2	0.118873	0	1	2	0.133462	0
	x4	5	51	1.457223	0	1	2	0.118314	0	1	2	0.12823	0
	x6	2	15	0.474047	0	43	45	4.310932	6.06E-11	39	41	4.837805	7.19E-11
	x7	1	3	0.098135	0	1	2	0.086115	0	1	2	0.109128	0
	x8	1	3	0.124726	0	1	2	0.113855	0	1	2	0.128006	0

Table 3.6: Summary of test results reported in Table Tables 3.2-3.5

Methods	NI	Percentage	FEV	Percentage	Time (s)	Percentage
CHCG	91	65.00%	68	48.57%	102	72.86 %
PCG	0	0.00%	25	17.86%	28	20.00%
ETTG	5	3.57%	0	0.00%	10	7.14%
Undecided	44	31.43%	47	33.57%	0	0.00%

From Tables 3.2-3.5 one can easily see that, the three methods are trying to solve (1.3), but the improvement of the proposed method is clear. For example, CHCG solve all the problems but ETTCG fails, this is quite clear with problem 1. In fact, the CHCG approach significantly outperforms the PCG and ETTCG methods for nearly all the problems assessed, since it has the least number of iterations, functions evaluations and CPU time, which are far below the number of iterations, functions evaluations and the CPU time for the PCG and ETTCG methods. This is apparently due to the hybridization of conjugate gradient iterative scheme, correction parameter, as well as Jacobian approximation via acceleration parameter at each iteration.

The results presented in Tables 3.2-3.5 are summaries in Tables 3.6 to illustrate which approach is the winner in terms of number of iterations, functions evaluations and CPU time. The summary shows that the CHCG method is the most efficient as it solves 65.00% (91 out of 140) of the problems with least number of iterations than PCG and ETTCG methods which solve 0.00% (0 out of 140) and 3.57% (5 out of 140) respectively. However, the summarized result shows that the two or three methods solve 44 out of 140 problems with the same number of iteration, which translates to 31.43% and is reported as undecided. The table also shows that the CHCG method solves 72.86% (102 out of 140) of the problems with less CPU time compared to PCG and ETTCG methods which solve 20.00% (28 out of 140) and 7.14% (10 out of 140) respectively. Moreover, for the function values, the proposed method solves 48.75% (68 out of 140), PCG method solves 17.86% (25 out of 140), and ETTCG method solves 0.00% (0 out of 140), and 33.57% (47 out of 140) is reported as undecided. Finally, from three Figures and the results in Tables Tables 3.2-3.5 it is obvious that our approach is very successful in solving large-scale nonlinear problems.

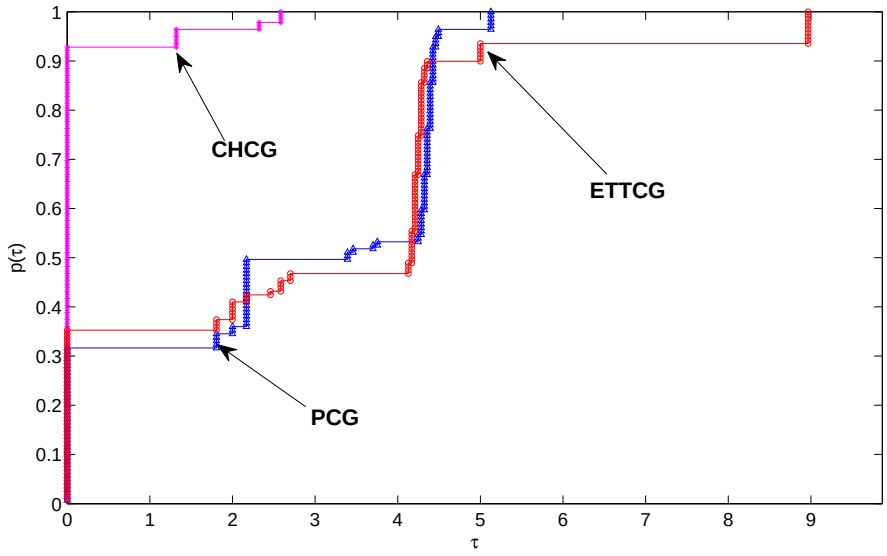

Figure 3.1: Performance profile for the number of iterations

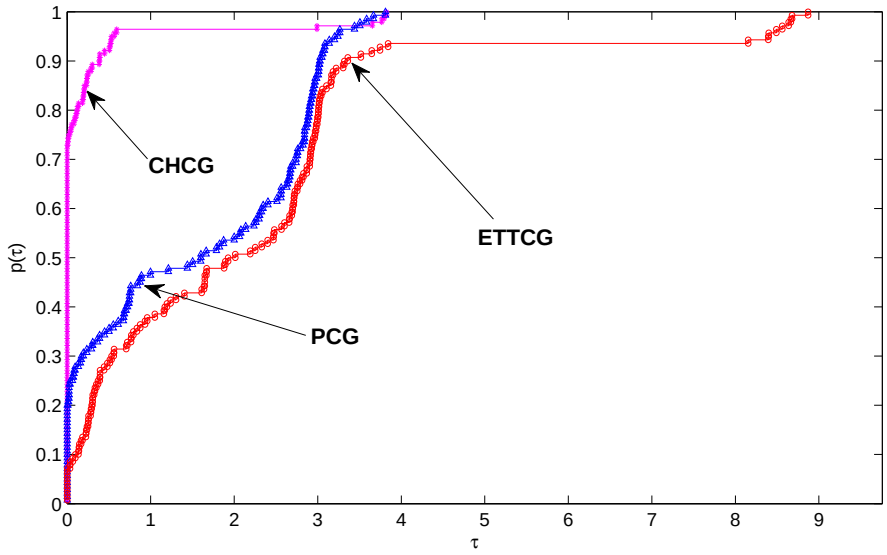

Figure 3.2: Performance profile for the CPU time (in second)

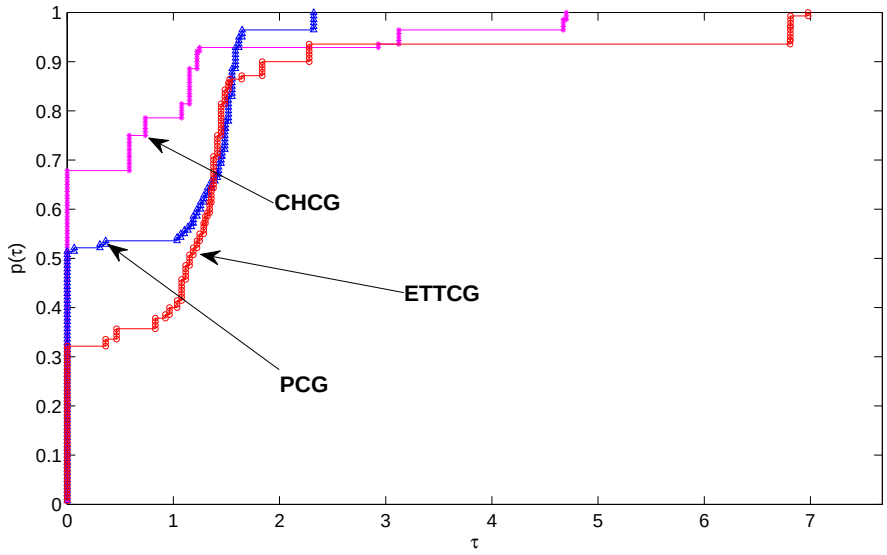

Figure 3.3: Performance profile for the functions evaluations

The results and summary presented in Tables Tables 3.2-3.6 are best described through a pictorial representation. To achieve this, we generate Figures 3.1-3.3 using the performance profiles of Dolan and Moré [44], which shows the performance of each of the three methods. For each problem $\rho \in \mathscr{P}$ and solver $s \in \mathscr{S}$, the performance profile is obtained in term of the performance measure $t_{\rho,s} > 0$. For any pair (ρ, s) of problem ρ and solver s, the performance ratio is given as

$$r_{\rho,s} = \frac{t_{\rho,s}}{\min\{t_{\rho,s} | s \in \mathscr{S}\}}. \tag{3.97}$$

The best solver for a particular problem reaches the lower bound $r_{\rho,s} = 1$. If a solver s fails to meet the convergence test for problem ρ, then $r_{\rho,s}$ is set to infinity. The performance profile of a solver s is defined as

$$p(\tau) = \frac{1}{n_\rho} \text{size}\{\rho \in \mathscr{P} | r_{\rho,s} \leq \tau\}, \tag{3.98}$$

where n_p is the number of problems. Therefore, $p(\tau)$ is the probability for solver $s \in \mathscr{S}$ that a performance ratio $r_{\rho,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio. However, the function P is the cumulative distribution function for the performance ratio. For this, we plot the fraction $p(\tau)$ of the problems for which each method is within τ of the smallest number of iterations, CPU time and function evaluations respectively.

Observe that, in Figures 3.1-3.3, the curves corresponding to the CHCG method remain above the other curves representing the PCG and ETTCG methods. This shows that the method proposed outperforms the methods compared in terms of fewer iterations, functions evaluation and CPU time (in seconds) and is therefore the most efficient method. It is vital to state here that the advantages the CHCG method had over the PCG and ETTCG methods includes

(i) the ability to converge much faster to solutions of the problems, which is shown by the final norm value attained for each problem,

(ii) better performance with uniform as well as mixed valued initial starting points.

The limitations of the CHCG method lies in the fact that if the correction parameter t_k is assigned outside the interval $(1,2)$, then it takes much time to converge and in some instances diverges.

3.5 Applications of CHCG Algorithm in compressive sensing

This subsection is dedicated to application of the proposed algorithm (CHCG) to signal recovery problems in compressive sensing. This was accomplished by solving the monotone nonlinear system of equations (2.45) which is the reformulation of the ℓ_1-norm regularization problems in compressive sensing (2.39). To further highlight the performance of the CHCG scheme, some numerical experiments were carried out by considering a typical compressive sensing problem. The main aim is to reconstruct a sparse signal of length-n from k observations.

The mean of squared error (MSE) to the original signal $\hat{x}$ is used to assess the quality of the restoration i.e.,

$$MSE = \frac{1}{n}\|\hat{x} - \tilde{x}\|^2, \tag{3.99}$$

where $\tilde{x}$ denotes the restored or recovered signal. In the experiment, the size of the signal is tested with $n = 2^{12}$ and $k = 2^{10}$. The original signal contains 2^7 randomly nonzero elements. Also, the measurement ϑ is distributed by noise, $\vartheta = M\hat{x} + \omega$ where ω is the Gaussian noise distributed as $N(0, 10^{-4})$ and M is the Gaussian matrix generated in Matlab by the command $randn(m,n)$.

To show the performance of the CHCG scheme in compressive sensing, we compare it with the effective PCG method [73]. We set the parameters for both methods as $\xi = 10$, $\sigma = 10^{-4}$ and $\rho = 0.5$, we however set the parameter $r = 0.2$ in PCG method. The merit function $f(x) = \frac{1}{2}\|\vartheta - Mx\|_2^2 + \pi\|x\|_1$, in the experiment. We run each code with the same initial point, use the same continuation technique on the π parameter and only observe each method's convergence behavior to get a solution with similar accuracy. In addition, the measuring signal begins the iterative process for the experiment, i.e, $x_0 = M^T\vartheta$ and terminate when the objective function's relative change satisfies

$$\frac{|f(x_k) - f(x_{k-1})|}{|f(x_{k-1})|} < 10^{-5}, \tag{3.100}$$

where $f(x_k)$ denotes the function value at x_k.

To highlight the performance of the proposed CHCG method, we test it against PCG solver that is applied to solve monotone equations and has application in compressive sensing. For both methods, we carried out ten (10) experiments for different noise samples, the average of the experiments is also computed and reported the results in Table 3.7.

Furthermore, from Figure 3.4, almost exactly the disturbed signal has been restored by CHCG and PCG methods. To show the performance of these methods visibly, we plot four graphs to exhibit the convergence behavior of the two methods through their mean square error (MSE) and function evaluations results, as well as the number of iterations and CPU time respectively. As can be seen from

Table 3.7: Ten numerical results of the ℓ_1-norm regularization problem for CHCG and PCG methods.

	CHCG			**PCG**		
	MSE	ITER	TIME(s)	MSE	ITER	TIME(s)
	1.05E-05	118	3.17	3.47E-05	146	4.50
	2.54E-05	119	3.63	5.19E-05	126	3.38
	1.72E-05	90	2.47	2.36E-05	125	3.31
	2.36E-05	102	3.81	2.27E-05	142	3.58
	3.42E-05	117	3.51	4.51E-05	144	3.75
	2.63E-05	122	3.22	5.54E-05	135	3.86
	2.14E-05	130	3.70	2.50E-05	147	3.72
	3.56E-05	139	3.81	5.56E-05	146	4.56
	2.61E-05	124	3.53	3.69E-05	153	4.13
	2.40E-05	114	3.33	3.03E-05	151	5.20
Average	2.44E-05	117.5	3.418	3.81E-05	141.5	3.999

Figure 3.5, our proposed method exhibits much faster descent rates of MSE and function evaluations than the PCG method. It can also be seen from the figures that the CHCG method requires less number of iterations and CPU time in order to recover the original signal compared to that required by the PCG method for the same process.

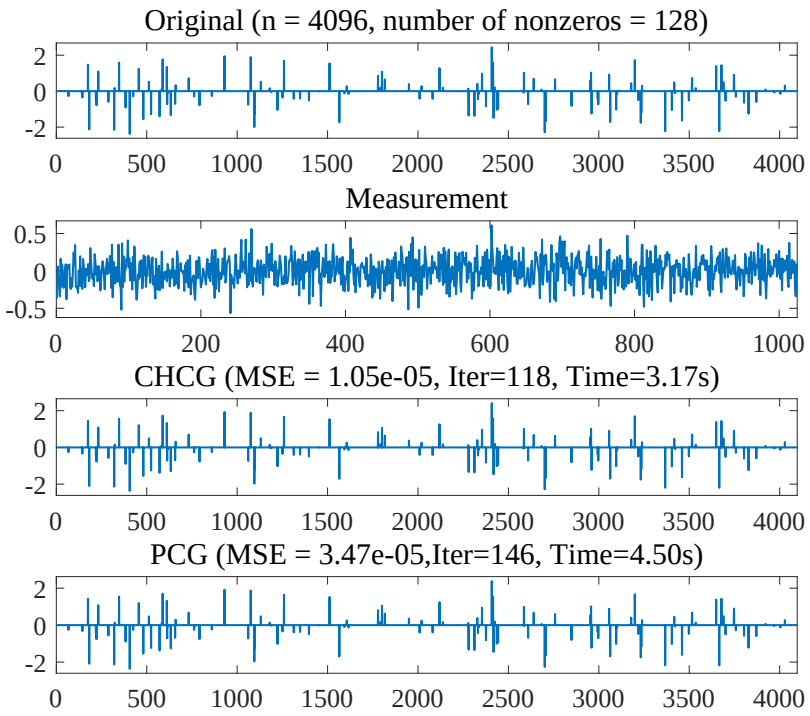

Figure 3.4: The original signal, the measurement, and the recovery signals using the CHCG and PCG methods are shown from top to bottom.

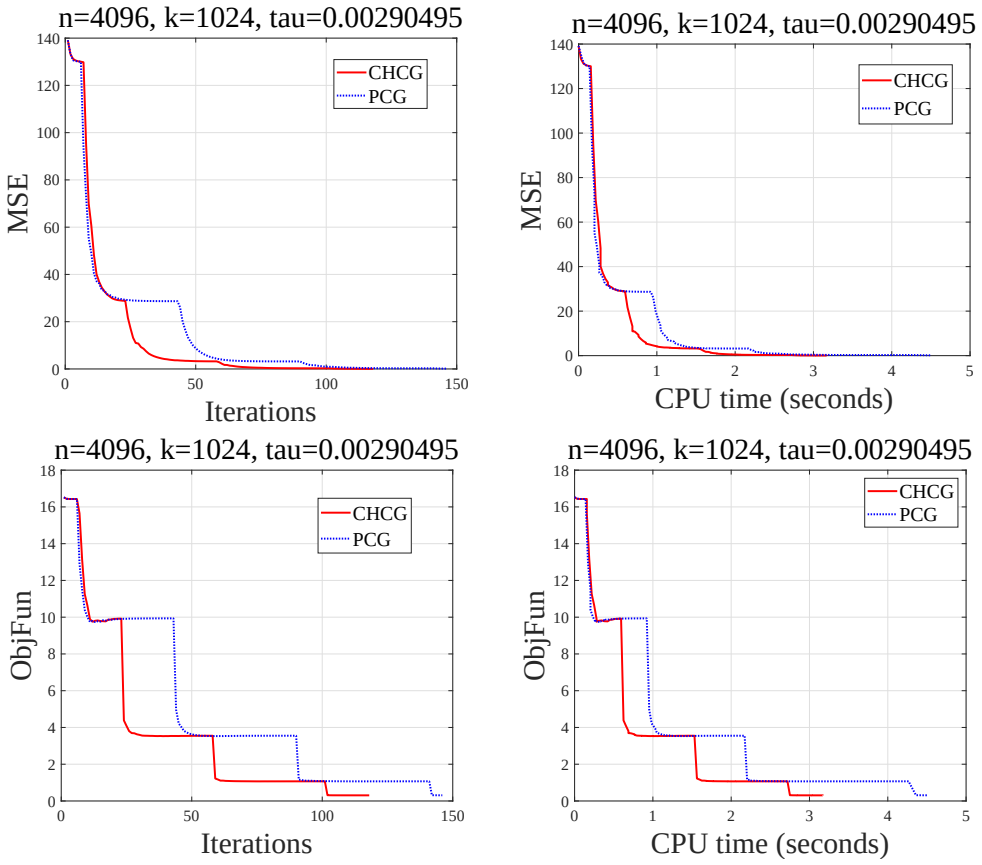

Figure 3.5: Comparison of the results of CHCG and PCG methods.

3.6 Conclusion

In this chapter, a convex constrained hybrid conjugate gradient method for monotone nonlinear equations with application to signal recovery is presented. This was achieved by combining the direction proposed in [59] with the classical conjugate gradient direction by applying Picard-Mann hybrid iterative process and derived an effective conjugate gradient parameter. It is a completely matrix-free method that is globally convergent under certain appropriate conditions. Numerical comparisons have been made using large-scale test problems. Furthermore, Tables and Figures showed that the proposed method is practically quite welcome, because it has the least number of iteration and CPU time compared to PCG [73] and ETTCG [52] methods respectively. In addition, the proposed method is successfully applied to deal with the experiments on the ℓ_1-norm regularization problem in compressive sensing, where ten (10) experiments are conducted with different samples of noise, and the experiments average values are also reported in Table 7 which clearly showed a better efficiency for CHCG method. Finally, Figure 5 indicated that the CHCG method had better restoration of the disturbed signal because it had the lowest mean square error (MSE) value, iteration number, and CPU time than the PCG method.

CHAPTER FOUR

Accelerated Dai-Liao Methods for Solving System of Nonlinear Equations

4.1 Introduction

Conjugate gradient method has been proved to be one of the most ideal iterative methods for handling large-scale system of monotone nonlinear equations due to their simple implementation, low storage requirement and global convergence properties. In this chapter, a conjugate gradient projection algorithm to solve convex constrained monotone nonlinear equations (1.3) is presented.

Newton's method is one of the most common approach to find a solution in (1.3), because of its nice properties, such as rapid convergence rate from a reasonably good starting point $x_0 \in \mathbb{R}^n$ in a neighborhood of the solution with search direction d_k given in (1.9). However, in Newton's method, the derivative F' is computed at each iteration, which may be unavailable or could not be obtained exactly. In this case Newton's method cannot be applied directly. For this reason, quasi-Newton's methods were developed to replace the Jacobian matrix or its inverse with an approximation which can be updated at each iteration [42, 125], and its search direction is given in (1.11)

Despite the appealing characteristics of the Newton and quasi-Newton's methods, the derivative F' or its approximation is computed at each iteration, so they are not ideal for solving large-scale problems because of their requirement for huge matrix storage at each iteration, which is costly in numerical experiments. To overcome these problems, Conjugate gradient methods are proposed [120, 4,

35, 50, 88], with the search direction given in (1.12). The proposed method is based on proposing new value of nonnegative parameter t in Dai-Liao Conjugate Gradient Method. This is made possible by combining the conjugate gradient method with the Newton method approach. The Jacobian matrix is approximated via acceleration parameter in order to solve the large-scale of problems. The proposed method is proven to be globally convergent under some mild conditions. Moreover, in order to demonstrate the new schemes efficiency, it is extended to solve general system of nonlinear equations. The numerical experiments, shown in this chapter, depicts the efficiency of the proposed methods. Throughout this chapter, the space $\mathbb{R}^n$ denote the $n-$dimensional real space equipped with the Euclidean norm $\|\cdot\|$, $F_k = F(x_k)$, and $F'_k = F'(x_k)$.

4.2 New choice of Dai-Liao nonnegative parameter

If a point x_k is sufficiently close to the solution say x^*, then Newton's direction in (1.9) is the one to follow. Therefore, from the CG direction in (1.12) and the Dai-Liao CG parameter in (2.35), the value of parameter t can be computed as follows:

$$-(F'_k)^{-1}F_k = -F_k + \frac{F_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} d_{k-1} - t\frac{F_k^T s_{k-1}}{d_{k-1}^T y_{k-1}} d_{k-1}. \tag{4.101}$$

With a view to avoiding the computation of the Jacobian matrix F'_k and its inverse, we are interested in approximating the Jacobian matrix via

$$F'_k \approx \gamma_k I, \tag{4.102}$$

where, γ_k is an acceleration parameter presented as

$$\gamma_k = \tau_k^{BB1}(\tau_k^{BB2})^{-1} = \frac{\|y_{k-1}\|^2 \|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2}, \tag{4.103}$$

where, τ_k^{BB1} and τ_k^{BB2} are in (2.15). From (1.9) and (4.102), the search direction is defined as

$$d_k = -\gamma_k^{-1} F_k. \tag{4.104}$$

By substituting (4.102), (4.103) and (4.104) in (4.101), and applying some algebraic calculation, the proposed value of the parameter t can be obtained as

$$t = \frac{(F_k^T s_{k-1})(y_{k-1}^T s_{k-1})^3 + (F_k^T y_{k-1})\|y_{k-1}\|^2\|s_{k-1}\|^4 - (F_k^T s_{k-1})(y_{k-1}^T s_{k-1})\|y_{k-1}\|^2\|s_{k-1}\|^2}{(F_k^T s_{k-1})\|y_{k-1}\|^2\|s_{k-1}\|^4}. \tag{4.105}$$

To ensure that our search direction satisfies the descent condition, we incorporated to the value of parameter $t \geq 0$ presented in [7]. Therefore the value of the proposed nonnegative parameter t denoted as $\bar{t}$ can be presented as

$$\bar{t} = \max\left\{t, \varphi \frac{\|y_{k-1}\|^2}{y_{k-1}^T s_{k-1}}\right\}. \tag{4.106}$$

where $\varphi \geq \frac{1}{4}$ and the proposed search direction d_k is given as

$$d_k = \begin{cases} -F_k, & \text{if} \quad k = 0, \\ -F_k + \frac{(y_{k-1} - \bar{t}s_{k-1})^T F_k}{d_{k-1}^T y_{k-1}} d_{k-1}, & \text{if} \quad k \geq 1, \end{cases} \tag{4.107}$$

Algorithm 2: Accelerated Dia-Liao projection Method (ADLP)

Input: Given $x_0 \in \Omega$, $d_0 = -F_0$, $\rho \in (0,1)$, $\sigma > 0$, $\varepsilon = 10^{-6}$, and $\xi > 0$, set $k = 0$.

Step 1: Compute F_k. If $\|F_k\| \leq \varepsilon$ then stop, else go to Step 2.

Step 2: Let $\alpha_k = \xi \rho^{m_k}$, with m_k being the smallest nonnegative integer m such that

$$-F(x_k + \alpha_k d_k)^T d_k \geq \sigma \alpha_k \|d_k\|^2. \tag{4.108}$$

Step 3: Set $z_k = x_k + \alpha_k d_k$.

Step 4: If $z_k \in \Omega$ and $\|F(z_k)\| \leq \varepsilon$, stop, otherwise go to Step 5.

Step 5: Compute the next iterate by

$$x_{k+1} = P_\Omega[x_k - \lambda_k F(z_k)], \tag{4.109}$$

where, $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step 6: Compute the search direction $d_k = -F_k + \frac{(y_{k-1} - \bar{t} s_{k-1})^T F_k}{d_{k-1}^T y_{k-1}} d_{k-1}$, where $\bar{t}$ is defined in (4.106).

Step 7: Consider $k = k+1$ and go to Step 2.

Remark 4.2.1 *We give the following remarks*

(a) From (3.50), we have

$$y_{k-1}^T s_{k-1} = s_{k-1}^T (F(z_{k-1}) - F(x_{k-1})) \geq c \|s_{k-1}\|^2 > 0. \tag{4.110}$$

From (3.49) and Cauchy-Schwarz inequality, we have

$$y_{k-1}^T s_{k-1} = (F(z_{k-1}) - F(x_{k-1}))^T (z_{k-1} - x_{k-1}) \leq L \|s_{k-1}\|^2. \tag{4.111}$$

Also, from (4.110) and (4.111) it is simple to check that, $L\|s_{k-1}\|^2 \geq y_{k-1}^T s_{k-1} \geq c\|s_{k-1}\|^2$, *this implies that* $\frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \leq \frac{1}{c}$ *and* $\frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \geq \frac{1}{L}$.

(b) From assumption (A1) we have

$$\|y_{k-1}\| = \|F(z_{k-1}) - F(x_{k-1})\| \leq L\|s_{k-1}\| = L\alpha_{k-1}\|d_{k-1}\|. \tag{4.112}$$

(c) *From assumption (A2) we have*

$$y_{k-1}^T d_{k-1} = (F(z_{k-1}) - F(x_{k-1}))^T \frac{s_{k-1}}{\alpha_{k-1}} \geq \frac{c\|s_{k-1}\|^2}{\alpha_{k-1}} = c\alpha_{k-1}\|d_{k-1}\|^2 > 0. \tag{4.113}$$

Since $y_{k-1}^T d_{k-1} > 0$, the proposed direction in (4.107) is well defined.

(d) *From assumption (A1) and (A2) we have*

$$\gamma_k = \frac{\|y_{k-1}\|^2\|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2} \leq \frac{L^2\|s_{k-1}\|^4}{c^2\|s_{k-1}\|^4} = \frac{L^2}{c^2}. \tag{4.114}$$

This means γ_k in (4.103) is well defined.
Consequently, we have

$$\frac{\|y_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \leq \frac{L^2\|s_{k-1}\|^2}{c\|s_{k-1}\|^2} = \frac{L^2}{c}. \tag{4.115}$$

Remark 4.2.2 *To establish the global convergence property of Algorithm 2, the proposed nonnegative $\bar{t}$ in (4.106) needs to be bounded. Therefore, we set*

$$\bar{t} \leftarrow \min\{\bar{t}, H\}, \tag{4.116}$$

where H is a suitable positive number.

4.3 Convergence Analysis of Algorithm 2 (ADLP)

In this subsection, the global convergence of Algorithm 2 is established.

Lemma 4.3.1 *Suppose assumptions (A1)-(A3) hold and $\{x_k\}$ be generated by Algorithm 2 with $t = \bar{t}$. Since (4.113) guarantees $y_{k-1}^T d_{k-1} > 0$, for all $k \geq 0$, then the search direction d_k satisfies the following property*

$$F_k^T d_k \leq -c^*\|F_k\|^2. \tag{4.117}$$

where $c^* = (1 - \frac{1}{4\varphi})$.

Proof For $k=0$, from (4.107) we have $F_0^T d_0 = -\|F_0\|^2 < 0$. For $k \geq 1$, we have,

$$\begin{aligned} F_k^T d_k &= F_k^T \left(-F_k + \frac{F_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} d_{k-1} - \bar{t} \frac{F_k^T s_{k-1}}{d_{k-1}^T y_{k-1}} d_{k-1} \right), \\ &= -\|F_k\|^2 + \frac{(F_k^T y_{k-1})(F_k^T s_{k-1})}{s_{k-1}^T y_{k-1}} - \bar{t} \frac{(F_k^T s_{k-1})^2}{s_{k-1}^T y_{k-1}}, \\ &\leq -\|F_k\|^2 + \frac{(F_k^T y_{k-1})(F_k^T s_{k-1})}{s_{k-1}^T y_{k-1}} - \varphi \frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}} \frac{(F_k^T s_{k-1})^2}{s_{k-1}^T y_{k-1}}, \\ &= F_k^T d_k^{\varphi}. \end{aligned} \tag{4.118}$$

■

From (4.118) it is clear that $F_k^T d_k \leq F_k^T d_k$. However, from Theorem 1 of [17], it has been proved that $F_k^T d_k^{\varphi} \leq c^* \|F_k\|^2$. Therefore, $F_k^T d_k \leq c^* \|F_k\|^2$. Hence our proposed direction satisfies the property in (4.117).

Lemma 4.3.2 *Suppose assumptions (A1)-(A3) hold and let $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm 2, then $\{x_k\}$ and $\{z_k\}$ are bounded. In addition, we have*

$$\|d_k\| \leq M. \tag{4.119}$$

$$\lim_{k\to\infty} \|x_k - z_k\| = 0. \tag{4.120}$$

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{4.121}$$

Proof The proofs of (4.120) and (4.121) follow from Lemma 3.3.3. As a result, there are some constants $m_1 > 0$ and $\kappa > 0$ such that

$$\|F(x_k)\| \leq m_1, \tag{4.122}$$

$$\|F(z_k)\| \leq \kappa, \tag{4.123}$$

and

$$\lim_{k\to\infty} \alpha_k \|d_k\| = 0. \tag{4.124}$$

Next, from (4.107),(4.110) and (4.116) we have

$$\begin{aligned}
\|d_k\| &= \| - F_k + \frac{(y_{k-1} - \bar{t}s_{k-1})^T F_k}{d_{k-1}^T y_{k-1}} d_{k-1} \| \\
&\leq \|F_k\| + \left| \frac{y_{k-1}^T F_k - \bar{t} s_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} \right| \|s_{k-1}\| \\
&\leq \|F_k\| + \left(\frac{\|y_{k-1}\| \|F_k\| + \bar{t}\|s_{k-1}\| \|F_k\|}{y_{k-1}^T s_{k-1}} \right) \|s_{k-1}\| \\
&\leq \|F_k\| + \left(\frac{(L\|s_{k-1}\| + \bar{t}\|s_{k-1}\|)\|F_k\|}{y_{k-1}^T s_{k-1}} \right) \|s_{k-1}\| \\
&\leq \|F_k\| + \left(\frac{(L+\bar{t})\|s_{k-1}\| \|F_k\|}{c\|s_{k-1}\|^2} \right) \|s_{k-1}\| \\
&\leq \left(1 + \frac{(L+H)}{c} \right) m_1. \qquad (4.125)
\end{aligned}$$

Where Cauchy–Schwarz inequality is applied in the second inequality.
Taking $M = \left(1 + \frac{(L+H)}{c}\right) m_1$ we have (4.119). ■

Theorem 4.3.3 *Suppose assumption (A1)-(A3) hold and* $\{x_k\}$ *be generated by Algorithm 2. Then*

$$\liminf_{k \to \infty} \|F(x_k)\| = 0. \qquad (4.126)$$

Proof For the sake of contradiction, suppose that (4.126) is not true, then there exists $n_0 > 0$ such that

$$\|F(x_k)\| \geq \delta_0 \quad \text{holds}, \quad \forall k > 0. \qquad (4.127)$$

Suppose that the search direction $\|d_k\| \neq 0$ unless at the solution, then there exist a constant, say δ_1

$$\|d_k\| \geq \delta_1. \qquad (4.128)$$

If $\alpha_k \neq \xi$, then by the definition of $\alpha_k = (\alpha_k + \alpha_k^2 \gamma_k)$, $\rho^{-1}\alpha_k$ does not satisfy (4.108) i.e.,

$$-F(x_k + \rho^{-1}\alpha_k d_k)^T d_k < \sigma \rho^{-1} \alpha_k \|d_k\|^2.$$

Now, combining with (4.118) gives

$$\begin{aligned} c^*\|F(x_k)\|^2 &\leq -F(x_k)^T d_k, \\ &= (F(x_k+\rho^{-1}\alpha_k d_k)-F(x_k))^T d_k - F(x_k+\rho^{-1}\alpha_k d_k)^T d_k, \\ &\leq L\rho^{-1}\alpha_k\|d_k\|^2+\sigma\rho^{-1}\alpha_k\|d_k\|^2. \\ &= \alpha_k\|d_k\|(L+\sigma)\rho^{-1}\|d_k\|. \end{aligned} \tag{4.129}$$

Therefore, from (4.129), we have

$$\alpha_k\|d_k\| > \frac{\rho}{(L+\sigma)}\frac{c^*\|F_k\|^2}{\|d_k\|} \geq \frac{\rho}{(L+\sigma)}\frac{(1-\frac{1}{4\varphi})\delta_0^2}{M}. \tag{4.130}$$

The inequality (4.130) contradicts (4.124). Therefore (4.126) holds. Hence, the proof is completed. ■

4.4 Numerical Experiments on monotone nonlinear equations with convex constraints

In this subsection, the numerical performance of the proposed method (ADLP) for solving some large-scale monotone system of nonlinear equations is presented. Since the proposed algorithms is based on the nonnegative parameter $t > 0$, therefore its performances is compared with the following methods in the existing literature.

- DLPM Algorithm proposed in [1] with $t = p\frac{\|y_{k-1}\|^2}{y_{k-1}^T s_{k-1}} - q\frac{y_{k-1}^T s_{k-1}}{\|s_{k-1}\|^2}$, $p \geq \frac{1}{4}$, $q \leq 0$.
- PGG Algorithm proposed in [73] with β_k^{HZ} i.e., the parameter $t = 2\frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}}$.

The computer codes used are written in Matlab 8.3.0 (R2014a) and run on a personal computer equipped with a 1.80 GHz CPU processor and 8 GB RAM. When implementing our algorithm (ADLP) in this experiments, the following parameters are set; $\xi = 1$, $\sigma = 10^{-4}$ and $\rho = 0.9$. The parameters of DLPM and PCG Algorithms, are taken as in [2] and [73] respectively. The iteration is set to stop for

all the three methods if the following conditions occur: (i) when $\|F(x_k)\| \leq 10^{-6}$, (ii) when $\|F(z_k)\| \leq 10^{-6}$, or (iii) when the iterations exceed 1000 but no point of x_k satisfying the stopping criterion is obtained. We have tried the three methods on the previous six test problems with different initial guess and dimension (n values). To show the extensive numerical experiments of ADLP, DLPM and PCG methods, the experimentation was carried out with three different dimensions namely, 1000, 50,000 and 100,000. The numerical results of the three (3) methods are reported in Tables 4.9-4.14. From the Tables,

- SP represents the starting points,
- NI represents the total number of iterations,
- TIME represents the CPU time (in seconds),
- FEV represents the functions evaluations,
- NORM is the norm of $\|F(x_k)\|$ at the termination point.

The following initial points were used for the test problems:

Table 4.8: Starting points used to test problems

Starting Points	Values
$x1$	$(1,1,...,1)^T$
$x2$	$\left(\frac{1}{2},\frac{1}{2^2},...,\frac{1}{2^n}\right)^T$
$x3$	$\left(0,\frac{1}{2},\frac{2}{3},...,1-\frac{1}{n}\right)^T$
$x4$	$\left(1,\frac{1}{2},\frac{1}{3},...,\frac{1}{n}\right)^T$
$x5$	$(2,2,...,2)^T$
$x6$	$\left(\frac{1}{4},\frac{-1}{4},...,\frac{(-1)^n}{4}\right)^T$

The following test problems were used in the experiments:

Problem 1 [12]

$F_1(x) = e^{x_1} - 1,$

$F_i(x) = e^{x_i} + x_{i-1} - 1, \quad i = 2,3,\ldots,n-1,$

where $\Omega = \mathbb{R}^n_+$.

Problem 2 [79]

$F_i(x) = log(x_i + 1) - \frac{x_i}{n}, \quad i = 2,3,\ldots,n-1,$

where $\Omega = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i > -1, \quad i = 1,2,\ldots,n\}$.

Problem 3 [1]

$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 4 [73]

$F_i(x) = e^{x_i} - 1, \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 5 [12]

$F_i(x) = 2x_i - \sin|x_i - 1|, \quad i = 1,2,\ldots,n,$

where $\Omega = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i > -1, \quad i = 1,2,\ldots,n\}$.

Problem 6 [73]

$F_1 = x_1 - e^{\cos\left(\frac{x_1+x_2}{n+1}\right)},$

$F_i = x_i - e^{\cos\left(\frac{x_{i-1}+x_i+x_{i+1}}{n+1}\right)},$

$F_n = x_n - e^{\cos\left(\frac{x_{n-1}+x_n}{n+1}\right)}, \quad i = 2,3,...,n-1,$

where $\Omega = \mathbb{R}^n_+$.

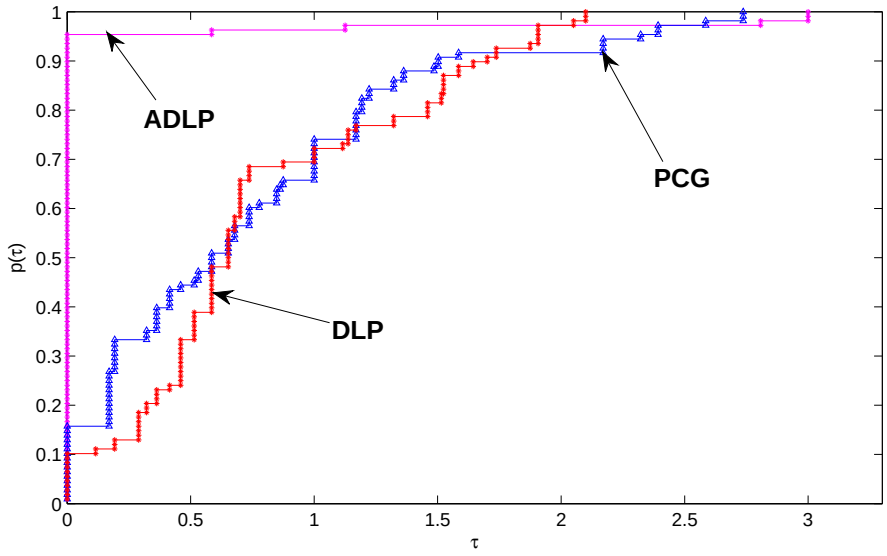

Figure 4.6: Performance profile for the number of iterations

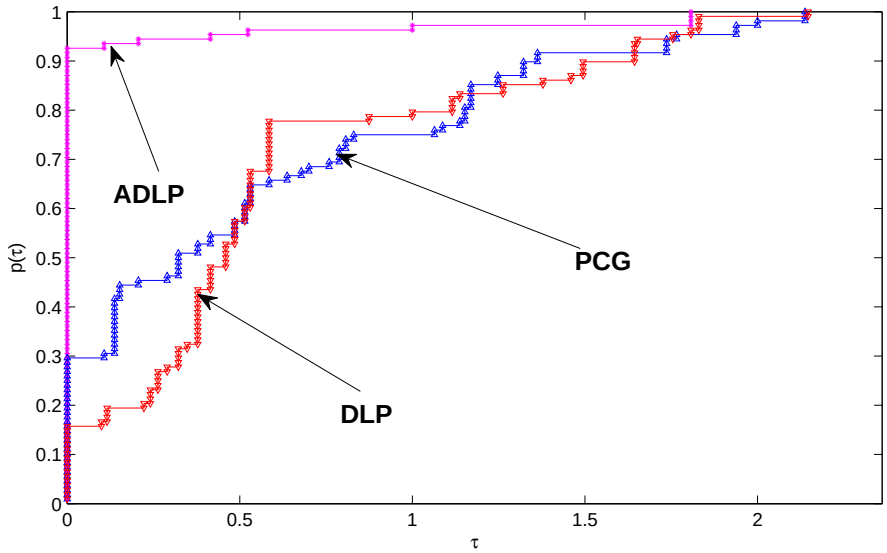

Figure 4.7: Performance profile for the functions evaluations

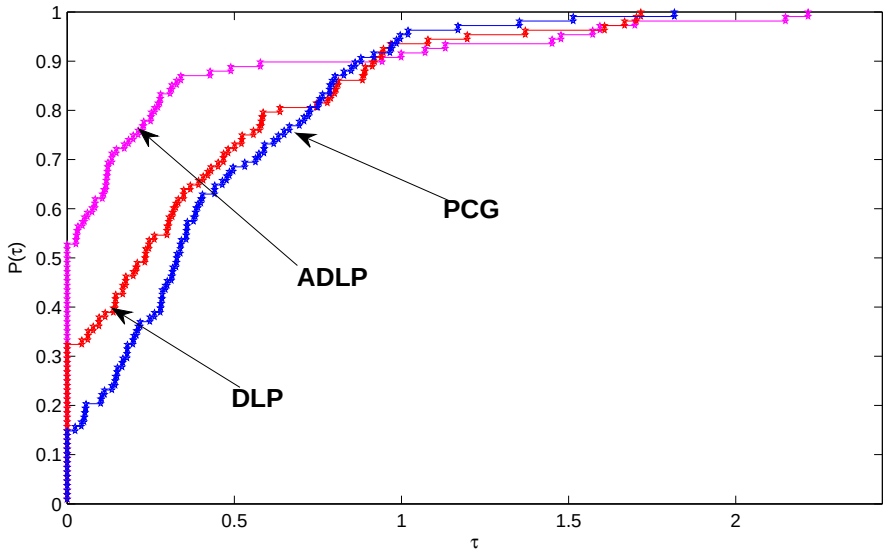

Figure 4.8: Performance profile for the CPU time (in second)

Table 4.9: Numerical results of ADLP, DLPM and PCG methods for problem 1

		ADLP				DLPM				PCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	5	7	0.012921	4.08E-07	16	18	0.024895	3.78E-07	20	22	0.029059	9.46E-07
	x2	9	11	0.012964	1.31E-07	18	19	0.013366	6.66E-07	30	31	0.016919	9.87E-07
	x3	7	8	0.011351	5.17E-08	16	18	0.009994	3.56E-07	20	22	0.015339	9.48E-07
	x4	8	9	0.012724	1.46E-07	18	20	0.010139	7.17E-07	30	32	0.017996	7.1E-07
	x5	6	7	0.006994	3.94E-07	16	18	0.012949	5.88E-07	29	31	0.019978	4.3E-07
	x6	5	7	0.008473	7.94E-07	1	2	0.002808	0	1	2	0.003873	0
50,000	x1	7	8	0.197687	1.1E-07	16	18	0.234369	5.75E-07	23	25	0.323135	5.95E-07
	x2	9	11	0.24295	1.31E-07	18	19	0.235325	6.66E-07	30	31	0.364638	9.87E-07
	x3	6	8	0.192067	1.15E-07	16	18	0.221475	5.98E-07	23	25	0.324235	5.95E-07
	x4	8	9	0.213488	1.47E-07	18	20	0.255809	7.22E-07	30	32	0.384726	7.11E-07
	x5	6	8	0.211875	1.3E-07	18	20	0.244994	5.03E-07	26	28	0.359808	5.3E-07
	x6	5	7	0.096581	7.36E-07	1	2	0.022754	0	1	2	0.021778	0
100,000	x1	7	8	0.318553	1.55E-07	16	18	0.411236	7.77E-07	23	25	0.578818	8.41E-07
	x2	9	11	0.45589	1.31E-07	18	19	0.420096	6.66E-07	30	31	0.772416	9.87E-07
	x3	7	8	0.338478	1.59E-07	16	18	0.401022	7.94E-07	23	25	0.661035	8.41E-07
	x4	8	9	0.375707	1.47E-07	18	20	0.462273	7.22E-07	30	32	0.743873	7.11E-07
	x5	7	8	0.342313	1.84E-07	18	20	0.473426	7.12E-07	25	27	0.671345	8.34E-07
	x6	5	7	0.175316	7.35E-07	1	2	0.037714	0	1	2	0.041869	0

Table 4.10: Numerical results of ADLP, DLPM and PCG methods for problem 2

		ADLP				DLPM				PCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	4	6	0.024961	3.6E-08	8	10	0.008966	3E-07	10	12	0.009612	2.23E-07
	x2	4	6	0.006512	5.55E-07	8	10	0.008009	9.21E-07	8	10	0.008101	6.98E-07
	x3	5	7	0.007504	3.52E-07	14	16	0.013067	7.89E-07	10	12	0.009113	2.25E-07
	x4	6	8	0.008879	2.73E-07	10	12	0.010454	1.76E-07	11	13	0.010286	1.15E-07
	x5	5	7	0.009336	1.74E-08	9	11	0.009973	3.31E-07	11	13	0.010569	2.76E-07
	x6	3	5	0.007286	1.01E-08	7	9	0.008229	1.62E-07	9	11	0.008166	1.88E-07
50,000	x1	5	7	0.110159	6.64E-07	9	11	0.164751	1.98E-07	11	13	0.219787	6.22E-07
	x2	7	9	0.127004	2.09E-07	8	10	0.161759	7.29E-07	8	10	0.160967	6.72E-07
	x3	8	10	0.185623	1.75E-07	12	14	0.232559	8.49E-07	11	13	0.215991	6.22E-07
	x4	8	10	0.165403	3.55E-07	10	12	0.182756	1.57E-07	11	13	0.216123	1.01E-07
	x5	6	8	0.133003	2E-07	10	12	0.185693	2.18E-07	12	14	0.231765	7.68E-07
	x6	7	9	0.161413	3.48E-07	8	10	0.157813	1.07E-07	10	12	0.199579	5.29E-07
100,000	x1	5	7	0.204172	9.14E-07	9	11	0.304295	2.8E-07	11	13	0.385612	8.79E-07
	x2	7	9	0.250392	2.09E-07	8	10	0.270916	7.27E-07	8	10	0.281085	6.71E-07
	x3	9	11	0.325695	2.78E-07	13	15	0.431866	5.25E-07	11	13	0.373266	8.79E-07
	x4	8	10	0.304361	3.55E-07	10	12	0.341622	1.53E-07	11	13	0.373209	1.01E-07
	x5	6	8	0.230353	2.69E-07	10	12	0.331095	3.08E-07	13	15	0.466981	1.01E-07
	x6	7	9	0.286812	4.92E-07	8	10	0.281027	1.51E-07	10	12	0.345831	7.47E-07

Table 4.11: Numerical results of ADLP, DLPM and PCG methods for problem 3

		ADLP				DLPM				PCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	7	9	0.014421	1.94E-07	7	9	0.004849	8.29E-07	10	12	0.008401	5.91E-07
	x2	5	7	0.006939	9.72E-07	8	10	0.006383	1.62E-07	8	10	0.008451	6.81E-07
	x3	7	9	0.006785	2.3E-07	10	12	0.008545	6.04E-07	10	12	0.008267	6.7E-07
	x4	6	8	0.006668	3.02E-07	11	13	0.008926	1.13E-07	9	11	0.008546	5.28E-07
	x5	7	9	0.008045	5.28E-07	7	9	0.005982	4.46E-07	10	12	0.006611	9.95E-07
	x6	1	2	0.003663	0	1	2	0.003684	0	1	2	0.002454	0
50,000	x1	8	10	0.120324	1.37E-07	8	10	0.099138	5.87E-07	11	13	0.134615	3.88E-07
	x2	5	7	0.081424	9.72E-07	8	10	0.150267	1.62E-07	8	10	0.101371	6.81E-07
	x3	8	10	0.107669	1.38E-07	11	13	0.141376	8.02E-07	11	13	0.129077	3.89E-07
	x4	6	8	0.083721	3.01E-07	12	14	0.163721	6.02E-08	9	11	0.109841	5.29E-07
	x5	8	10	0.117013	3.73E-07	8	10	0.096943	3.15E-07	12	14	0.143259	2.85E-07
	x6	1	2	0.040902	0	1	2	0.021357	0	1	2	0.020461	0
100,000	x1	8	10	0.201694	1.94E-07	8	10	0.179134	8.29E-07	11	13	0.258948	5.48E-07
	x2	5	7	0.157144	9.72E-07	8	10	0.184656	1.62E-07	8	10	0.204231	6.81E-07
	x3	8	10	0.215678	1.94E-07	12	13	0.266524	8.77E-07	11	13	0.236681	5.49E-07
	x4	6	8	0.166041	3.01E-07	12	14	0.308203	9.18E-07	9	11	0.208118	5.29E-07
	x5	8	10	0.218472	5.28E-07	8	10	0.182223	4.46E-07	12	14	0.271226	4.03E-07
	x6	1	2	0.088144	0	1	2	0.040228	0	1	2	0.044434	0

Table 4.12: Numerical results of ADLP, DLPM and PCG methods for problem 4

		ADLP				DLPM				PCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	7	9	0.017291	1.6E-07	8	10	0.005328	1.55E-07	10	12	0.006838	1.54E-07
	x2	7	8	0.005865	1.48E-07	9	10	0.006583	4.74E-07	9	10	0.006083	6.78E-07
	x3	8	9	0.007891	3.04E-07	12	13	0.009017	6.42E-07	10	12	0.007251	2.67E-07
	x4	9	10	0.007526	3.37E-07	12	14	0.006495	6.45E-07	11	13	0.006978	4.51E-07
	x5	6	8	0.007236	3.68E-07	8	10	0.008149	1.25E-07	10	12	0.007105	1.79E-07
	x6	1	2	0.001959	0	1	2	0.003689	0	1	2	0.006902	0
50,000	x1	8	10	0.108196	1.13E-07	9	11	0.097681	1.1E-07	11	13	0.122656	4.75E-07
	x2	7	8	0.080586	1.48E-07	9	10	0.089071	4.74E-07	9	10	0.083808	6.78E-07
	x3	9	10	0.096003	1.15E-07	15	16	0.167788	1.96E-07	11	13	0.116747	4.85E-07
	x4	9	10	0.123132	3.37E-07	12	14	0.151212	5.38E-07	11	13	0.11222	1.98E-07
	x5	7	9	0.115771	2.6E-07	8	10	0.091445	8.87E-07	11	13	0.113512	5.51E-07
	x6	1	2	0.013512	0	1	2	0.014263	0	1	2	0.013919	0
100,000	x1	8	10	0.195953	1.6E-07	9	11	0.178577	1.55E-07	11	13	0.205042	6.72E-07
	x2	7	8	0.132501	1.48E-07	9	10	0.155908	4.74E-07	9	10	0.153001	6.78E-07
	x3	9	10	0.181759	1.61E-07	12	14	0.224358	9.56E-08	11	13	0.196447	6.79E-07
	x4	9	10	0.199019	3.37E-07	13	15	0.275577	1.62E-08	11	13	0.191986	1.98E-07
	x5	7	9	0.187866	3.68E-07	9	11	0.177259	1.25E-07	11	13	0.214988	7.8E-07
	x6	1	2	0.025279	0	1	2	0.027016	0	1	2	0.026249	0

Table 4.13: Numerical results of ADLP, DLPM and PCG methods for problem 5

		ADLP				DLPM				PCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	3	5	0.013632	4.04E-08	18	20	0.012385	4.34E-07	9	11	0.010968	3.03E-07
	x2	6	8	0.009012	1.77E-08	15	17	0.013773	6.7E-07	9	11	0.007279	5.97E-07
	x3	6	8	0.010968	5.16E-07	18	20	0.013964	4.3E-07	9	11	0.013022	5.76E-07
	x4	9	11	0.012089	1.74E-08	21	23	0.015481	8.01E-07	12	14	0.010338	3.88E-07
	x5	4	6	0.008282	6.55E-08	18	20	0.017169	9.63E-07	9	11	0.008124	6.61E-07
	x6	3	5	0.009545	4.21E-07	15	17	0.014839	9.78E-07	10	12	0.010583	1.03E-07
50,000	x1	3	5	0.102841	2.86E-07	20	22	0.337909	4.75E-07	10	12	0.177207	9.02E-07
	x2	6	8	0.180204	6.2E-08	17	19	0.316167	4.23E-07	10	12	0.183229	2.2E-07
	x3	8	10	0.230792	7.43E-08	20	22	0.365047	4.75E-07	10	12	0.190355	9.3E-07
	x4	12	14	0.331518	4.82E-08	18	20	0.334163	5.3E-07	13	15	0.236112	8.94E-07
	x5	4	6	0.115354	4.63E-07	21	23	0.351527	4.15E-07	11	13	0.190991	1.33E-07
	x6	4	6	0.139681	1.49E-08	18	20	0.320272	4.21E-07	10	12	0.178868	7.26E-07
100,000	x1	3	5	0.207006	4.04E-07	20	22	0.657668	6.72E-07	11	13	0.356825	8.61E-08
	x2	6	8	0.326674	2.22E-07	17	19	0.564364	5.92E-07	10	12	0.338282	3.09E-07
	x3	24	26	1.062494	1.65E-08	20	22	0.651485	6.72E-07	11	13	0.388621	8.74E-08
	x4	21	23	1.081267	3.17E-07	18	20	0.641561	7.59E-07	14	16	0.514622	9.82E-08
	x5	4	6	0.216509	6.55E-07	21	23	0.704941	5.86E-07	11	13	0.350509	1.88E-07
	x6	4	6	0.231556	2.11E-08	18	20	0.598748	5.96E-07	11	13	0.389335	4.33E-07

Table 4.14: Numerical results of ADLP, DLPM and PCG methods for problem 6

		ADLP				DLPM				PCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	7	9	0.019575	5.41E-07	11	13	0.011265	1.94E-07	11	13	0.010186	2.45E-07
	x2	7	9	0.008185	8.56E-07	11	13	0.014041	6.93E-07	11	13	0.020912	3.88E-07
	x3	7	9	0.007956	5.43E-07	11	13	0.010883	2.07E-08	11	13	0.010183	2.46E-07
	x4	7	9	0.009149	8.54E-07	14	16	0.012656	3.63E-08	11	13	0.010363	3.87E-07
	x5	7	9	0.008629	2.26E-07	8	10	0.009499	5.00E-07	11	13	0.010511	1.02E-07
	x6	7	9	0.008629	9.35E-07	12	14	0.012699	8.09E-07	11	13	0.009778	4.24E-07
50,000	x1	8	10	0.201035	3.84E-07	9	11	0.175314	1.54E-07	12	14	0.247366	7.6E-07
	x2	8	10	0.193469	6.08E-07	9	11	0.179237	2.43E-07	13	15	0.243447	1.12E-07
	x3	8	10	0.191293	3.84E-07	9	11	0.180612	1.53E-07	12	14	0.226678	7.6E-07
	x4	8	10	0.194785	6.08E-07	9	11	0.179069	2.43E-07	13	15	0.249243	1.12E-07
	x5	8	10	0.191531	1.61E-07	8	10	0.163393	6.42E-07	11	13	0.206628	7.28E-07
	x6	8	10	0.201201	6.64E-07	9	11	0.192903	2.65E-07	13	15	0.254529	1.22E-07
100,000	x1	8	10	0.338761	5.43E-07	9	11	0.321665	2.17E-07	13	15	0.483794	9.97E-08
	x2	8	10	0.352989	8.6E-07	9	11	0.328128	3.44E-07	13	15	0.459372	1.58E-07
	x3	8	10	0.348595	5.43E-07	9	11	0.321712	2.17E-07	13	15	0.484483	9.97E-08
	x4	8	10	0.343856	8.6E-07	9	11	0.317002	3.44E-07	13	15	0.497208	1.58E-07
	x5	8	10	0.348567	2.27E-07	8	10	0.293118	9.09E-07	12	14	0.464829	4.49E-07
	x6	8	10	0.353994	9.39E-07	9	11	0.297269	3.75E-07	13	15	0.490187	1.72E-07

Table 4.15: Summary of test results reported in Table 4.9-4.14

Methods	NI	Percentage	FEV	Percentage	Time	Percentage
ADLP	89	82.41%	71	65.74%	57	52.78 %
DLPM	0	0.00%	0	0.00%	35	32.41%
PCG	2	1.85%	5	4.63%	16	14.81%
Undecided	17	15.74%	32	29.63%	0	0.00%

The results presented in Tables 4.9-4.14 clearly indicates that all the problems involved in the experiments were successfully solved by ADLP, DLP and PCG methods. However, ADLP method competes well with DLP and PCG methods as evident in the tables, the number of iteration recorded by ADLP is less than that of DLP and PCG methods. Therefore, the proposed method converges faster to the solution of (1.1). Notwithstanding, it can also be seen from the Tables 4.9-4.14 above, ADLP method has the least number of function evaluation than the compared methods. Furthermore, the numerical results reported in Tables **??**-5.26 evidently showed that ADLP method solved all the test problems with least CPU time. In fact, the ADLP approach significantly outperforms the DLP and PCG methods for nearly all the test problems assessed, since it has the least number of iterations, number of function evaluations and CPU time, compared to the two existing methods, that is, DLP and PCG.

To illustrate the best approach in terms of the number of iterations and processing time, Table 4.15 summarizes the results reported in Tables 4.9-4.14. From Table 4.15, the ADLP method wins least number of iteration in about 82.41% of the experiments (that is, 89 out of 108). While the DLP method did not win least number of iteration in the entire experiment and PCG method wins least number of iteration in only 1.85% (that is, 2 out of 108). Meanwhile, the summary of the results reveals that at least two of the three methods have the same number of iterations on 17 problems, which is 15.74% of the experiment (these are stated as undecided in the Table 4.15). For the processing time, the methods DLP and PCG have solved 32.41% (35 out of 108) and 14.81% (16 out of 108) respectively while the ADLP method has solved 52.78% (57 out of 108) of the problems with less CPU time. Moreover, for the number of function evaluations, the proposed method have solved 65.74% (71 out of 108), DLP method has solved 0.0% (0 out of 108), and PCG method has solved 4.63% (5 out of 108), where undecided is 29.63% (32 out of 108).

In order to interpret the results presented in Tables 4.9-4.14, Dolan and Mor*é* [44] evaluation tool was adopted to present a graphical view of each of the three methods involved in the experiments. By using the data from Tables 4.9-4.14 and the strategy proposed in [44], Figures 4.6-4.8 show the performance profiles of the ADLP, DLP and PCG methods regarding iteration numbers, number of function

evaluations and CPU time respectively. For each of the three figures, a fraction p(τ) of the problems considered is plotted for which a method is within a factor τ of the best time. The top curve in each of the three figures, is clearly corresponds to the one representing the ADLP scheme. Hence, the ADLP method is more efficient than the other methods in terms of least iteration numbers, function evaluations and processing time. Therefore, based on the results from Tables 4.9-4.14 and Figures 4.6-4.8, we conclude that the proposed algorithm is more efficient than the compared ones for solving large-scale nonlinear monotone equations with convex constraint.

4.5 Numerical Experiments on General System of Nonlinear Equations

In this experiment, the performance of the ADLP method have been demonstrated by using some benchmark general nonlinear equations to find a solution of (1.1). To show the effectiveness of our method, it is compared with the following existing methods in the literature.

- IDFDD Algorithm proposed in [59].
- ADLCG Algorithm proposed in [113].

Now, consider the proposed CG direction defined in (2.1). For general function F, it may be undefined at some iteration when its denominator $d_{k-1}^T y_{k-1}$ becomes zero. For this reason, we propose the following modified CG direction

$$d_k = -F_k + \frac{(w_{k-1} - \hat{t}s_{k-1})^T F_k}{d_{k-1}^T \psi_{k-1}} d_{k-1}. \tag{4.131}$$

$$\hat{t} = \max\left\{t, \varphi \frac{\|w_{k-1}\|^2}{\psi_{k-1}^T s_{k-1}}\right\}. \tag{4.132}$$

where, $\varphi \geq \frac{1}{4}$. Here,

$$\psi_{k-1} = w_{k-1} + \left(1 + \max\left\{0, -\frac{d_{k-1}^T w_{k-1}}{\|d_{k-1}\|^2}\right\}\right) d_{k-1}, \quad w_{k-1} = y_{k-1} + rs_{k-1}.$$

and $s_{k-1} = z_k - x_k = \alpha_k d_k$.

Remark 4.5.1 *We give the following remark.*
If $\max\left\{0, -\frac{d_{k-1}^T w_{k-1}}{\|d_{k-1}\|^2}\right\} \neq 0$ *in the definition of* ψ_{k-1}, *then*

$$\begin{aligned} d_{k-1}^T \psi_{k-1} &= d_{k-1}^T \left(w_{k-1} + d_{k-1} - \frac{d_{k-1}^T w_{k-1}}{\|d_{k-1}\|^2} d_{k-1} \right) \\ &= d_{k-1}^T w_{k-1} + \|d_{k-1}\|^2 - \frac{d_{k-1}^T w_{k-1}}{\|d_{k-1}\|^2} \|d_{k-1}\|^2 \\ &= \|d_{k-1}\|^2 > 0. \end{aligned} \tag{4.133}$$

Otherwise,

$$\begin{aligned} d_{k-1}^T \psi_{k-1} &= d_{k-1}^T w_{k-1} + \|d_{k-1}\|^2 \\ &= d_{k-1}^T (F_k - F_{k-1} + r d_{k-1}) + \|d_{k-1}\|^2 \\ &= d_{k-1}^T (F_k - F_{k-1}) + r\|d_{k-1}\|^2 + \|d_{k-1}\|^2 \\ &\geq (r+1)\|d_{k-1}\|^2 > 0. \end{aligned} \tag{4.134}$$

Since the function F *is monotone, the last inequality follows. This shows that* $d_{k-1}^T \psi_{k-1} > 0$ *provided that the solution is not reached. Therefore, the modified CG direction in* (4.131) *is well-defined.*

Furthermore, We use the derivative-free line search proposed by Li and Fukushima [69] in order to compute our step length α_k.

Let $f(x) = \frac{1}{2}\|F(x)\|^2$, $\omega_1 > 0$, $\omega_2 > 0$ and $r \in (0,1)$ be constants and let $\{\eta_k\}$ be a given positive sequence such that

$$\sum_{k=0}^{\infty} \eta_k < \eta < \infty \tag{4.135}$$

$$f(x_k + \alpha_k d_k) - f(x_k) \leq -\omega_1 \|\alpha_k F(x_k)\|^2 - \omega_2 \|\alpha_k d_k\|^2 + \eta_k f(x_k). \tag{4.136}$$

Let i_k is the smallest non negative integer i such that (4.136) holds for $\alpha = r^i$. Let $\alpha_k = r^{i_k}$.

Algorithm 3: Accelerated Dai-Liao Method(ADLM)

Input: Given $x_0 \in \mathbb{R}^n$, $\varepsilon = 10^{-6}$, and set $k = 0$.

Step 1: Compute F_k.

Step 2: If $\|F_k\| \leq \varepsilon$ then stop, else go to Step 3.

Step 3: Compute step the length α_k using (4.136).

Step 4: Set $x_{k+1} = x_k + \alpha_k d_k$.

Step 5: Compute the search direction $d_k = -F_k + \frac{(w_{k-1} - \hat{t}s_{k-1})^T F_k}{d_{k-1}^T \psi_{k-1}} d_{k-1}$, where $\hat{t}$ is defined in (4.132).

Step 6: Consider $k = k+1$ and go to Step 2.

When implementing ADLP in this experiments, the following parameters are set; $\omega_1 = \omega_2 = 10^{-4}$, $r = 0.2$ and $\eta_k = \frac{1}{(k+1)^2}$. The parameters of IDFDD and ADLCG Algorithms, are taken as in [59] and [113] respectively. The iteration is set to stop for all the three methods if the following conditions occur: (i) when $\|F(x_k)\| \leq 10^{-6}$ or (ii) when the iterations exceed 1000 but no point of x_k satisfying the stopping criterion is obtained. Similar to the experiment in Subsection 4.4, Problem 7–12 were solved using the same initial points in Table 4.8 with different dimensions. However, the initial point $x1$ is the solution of Problems 9 and 10. This means these two problems (i.e. 9 and 10) were solved using the remaining initial points in Table 4.8, that is, $x2 - x6$. The details of the experiment in this Subsection are reported in Tables 4.16–4.21. To show the extensive numerical experiments of ADLP, IDFDD and ADLCG methods, the experimentation was carried out with the dimensions between 100 to 10,000. The following test problems were used in the experiments:

Problem 7 [64]

$F_i(x) = x_i - 3x_i \left(\frac{\sin x_i}{3} - 0.66\right) + 2, \quad i = 1, 2, ..., n.$

Problem 8 [64] The discretized Chandrasehar H-equation (the problem of integral equation arising in radiative heat transfer)

$F_i(x) = x_i - \left(1 - \frac{c}{2n} \sum_{j=1}^{n} \frac{\mu_i x_j}{\mu_i + \mu_j}\right)^{-1}$, i=1,2,...,n, j=1,2,...,n.

with $c \in [0, 1)$ and $\mu = \frac{i-0.5}{n}$. (In our experiment we take c =0.1).

Problem 9 [59]

$F_i(x) = (1-x_i^2) + x_i(1 + x_i x_{n-2} x_{n-1} x_n) - 2, \quad i = 1,2,...,n.$

Problem 10 [115]

$F_1(x) = 3x_1^3 + 2x_2 - 5 + \sin(x_1 - x_2)\sin(x_1 + x_2),$

$F_i(x) = -x_{i-1}e^{x_{i-1}-x_i} + x_i(4 + 3x_i^2) + 2x_{i+1} + \sin(x_i - x_{i+1})\sin(x_i + x_{i+1}) - 8,$

$F_n(x) = -x_{n-1}e^{x_{n-1}-x_n} + 4x_n - 3, \quad i = 1,2,...,n.$

Problem 11 [113]

$F(x) = Ax + a,$

where, $A = \begin{pmatrix} 2 & -1 & & & \\ 0 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}$, and $a = (e_1^x - 1, ..., e_n^x - 1)^T$.

Problem 12 [59]

$F(x) = Bx + b_2,$

where, $B = \begin{pmatrix} 2 & -1 & & & \\ 0 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}$, and $b_2 = (\sin x_1 - 1, ..., \sin x_n - 1)^T$.

Table 4.16: Numerical results of ADLP, IDFDD and ADLCG methods for problem 7

		ADLP				IDFDD				ADLCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	15	31	0.041432	8.69E-07	42	85	0.042911	6.41E-07	13	27	0.034138	8.11E-07
	x2	13	27	0.015442	9.82E-07	38	77	0.025235	7.27E-07	12	25	0.01474	5.24E-07
	x3	15	31	0.011835	5.19E-07	41	83	0.019657	9.97E-07	13	27	0.011701	8.03E-07
	x4	13	27	0.010929	9.21E-07	38	77	0.018849	7.57E-07	12	25	0.011606	5.4E-07
	x5	16	33	0.014828	5.28E-07	43	87	0.023997	7.42E-07	15	30	0.010938	2.65E-07
	x6	13	27	0.010587	6.64E-07	36	73	0.019787	6.56E-07	12	25	0.009125	2.28E-07
10,000	x1	16	33	0.061181	6.86E-07	44	89	0.137224	8.31E-07	14	29	0.057689	5.15E-07
	x2	14	29	0.056426	9.86E-07	40	81	0.122641	9.36E-07	13	27	0.050256	3.31E-07
	x3	16	33	0.057493	5.97E-07	44	89	0.138927	8.31E-07	14	29	0.05685	5.14E-07
	x4	14	29	0.057935	7.8E-07	40	81	0.125207	9.41E-07	13	27	0.051634	3.32E-07
	x5	17	35	0.067829	4.16E-07	45	91	0.13749	9.61E-07	15	30	0.057664	8.37E-07
	x6	14	29	0.055652	5.24E-07	38	77	0.126101	8.51E-07	12	25	0.079804	7.21E-07
100,000	x1	17	35	0.431718	5.41E-07	47	95	0.970395	6.89E-07	15	31	0.690204	3.26E-07
	x2	15	31	0.38648	9.46E-07	43	87	0.902452	7.76E-07	14	29	0.358128	2.1E-07
	x3	17	35	0.442712	5.26E-07	47	95	1.018985	6.89E-07	15	31	0.604928	3.26E-07
	x4	15	31	0.398245	6.31E-07	43	87	0.905818	7.77E-07	14	29	0.33546	2.1E-07
	x5	18	37	0.446729	3.28E-07	48	97	0.993005	7.97E-07	16	32	0.59923	5.31E-07
	x6	15	31	0.389133	4.13E-07	41	83	0.836178	7.05E-07	13	27	0.477011	4.57E-07

Table 4.17: Numerical results of ADLP, IDFDD and ADLCG methods for problem 8

		ADLP				IDFDD				ADLCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x1	11	21	0.010937	6.75E-07	1000	1018	0.339969	0.268015	108	167	0.056498	6.71E-07
	x2	20	37	0.018584	5.53E-07	282	333	0.130165	7.77E-07	64	80	0.038859	6.7E-07
	x3	18	34	0.017121	6.46E-07	1000	1018	0.348101	0.266009	101	154	0.053857	6.1E-07
	x4	16	30	0.014891	8E-07	141	194	0.052984	8.1E-07	73	98	0.054086	6.04E-07
	x5	13	25	0.01493	2.15E-07	1000	1021	0.347612	0.227472	104	160	0.067722	6.36E-07
	x6	11	21	0.010739	4.45E-07	1000	1013	0.31116	0.263308	102	155	0.062945	6.02E-07
10,000	x1	13	25	0.067818	8.77E-07	1000	1031	2.546345	0.023955	107	165	0.638146	6.14E-07
	x2	13	23	0.059487	9.65E-07	116	166	0.336086	8.34E-07	60	71	0.213617	6.69E-07
	x3	14	27	0.062521	8.04E-07	1000	1031	2.289643	0.023931	107	165	0.357861	6.32E-07
	x4	10	18	0.044797	3.89E-07	110	163	0.321449	8.13E-07	79	111	0.272167	7.02E-07
	x5	14	27	0.065497	1.14E-07	1000	1034	2.551641	0.020336	110	171	0.367713	6.49E-07
	x6	13	25	0.059816	1.67E-07	1000	1026	2.547874	0.02353	108	166	0.348781	6.17E-07
100,000	x1	15	29	0.494577	1.88E-07	1000	1044	21.14728	0.002131	100	150	2.564255	9.98E-07
	x2	13	24	0.444689	9.58E-07	101	151	2.498897	8.34E-07	65	82	1.570939	7E-07
	x3	15	29	0.488453	1.9E-07	1000	1044	21.24248	0.002131	112	172	2.896463	9.59E-07
	x4	9	16	0.309649	7.55E-07	104	157	2.595913	8.13E-07	80	113	2.027689	6.53E-07
	x5	15	29	0.507944	8.98E-08	1000	1047	21.08481	0.001806	108	168	2.795744	7.89E-07
	x6	14	27	0.489574	1.73E-07	1000	1039	21.00534	0.002093	99	161	2.57339	7.05E-07

Table 4.18: Numerical results of ADLP, IDFDD and ADLCG methods for problem 9

		ADLP				IDFDD				ADLCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x2	15	28	0.013299	6.72E-07	52	95	0.026077	9.98E-07	12	21	0.009138	6.61E-07
	x3	14	25	0.011706	7.47E-07	79	131	0.035878	8.68E-07	18	31	0.012428	5.47E-07
	x4	14	25	0.01149	6.73E-07	48	86	0.021014	9.17E-07	24	42	0.0148	5.75E-07
	x5	14	30	0.012259	9.88E-07	40	82	0.020825	6.44E-07	13	28	0.009645	3.68E-07
	x6	13	27	0.011629	6.01E-07	38	77	0.017624	6.77E-07	11	22	0.008452	3.01E-07
10,000	x2	12	21	0.035889	2.94E-07	47	85	0.124418	7.27E-07	10	18	0.029038	9.97E-07
	x3	10	18	0.032377	8.51E-07	78	128	0.183987	9.41E-07	26	45	0.078171	8.05E-07
	x4	10	17	0.034121	6.74E-07	52	93	0.128158	8.56E-07	13	23	0.04141	2.19E-07
	x5	15	32	0.045694	7.81E-07	42	86	0.116847	8.34E-07	14	30	0.042155	2.33E-07
	x6	14	29	0.047751	4.75E-07	40	81	0.100142	8.77E-07	11	22	0.035756	9.52E-07
100,000	x2	11	21	0.289927	9.67E-07	54	98	1.075537	8.44E-07	7	13	0.195048	3.51E-07
	x3	10	18	0.247532	5.99E-07	80	132	1.524417	9.07E-07	14	25	0.333268	8.15E-07
	x4	10	17	0.243041	9.07E-07	53	95	1.083228	8.05E-07	13	22	0.324154	8.97E-08
	x5	16	34	0.400523	6.17E-07	45	92	0.93431	6.91E-07	14	30	0.351177	7.36E-07
	x6	15	31	0.388901	3.75E-07	43	87	0.903736	7.27E-07	12	24	0.297365	6.02E-07

Table 4.19: Numerical results of ADLP, IDFDD and ADLCG methods for problem 10

		ADLP				IDFDD				ADLCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
1000	x2	32	93	0.049091	5.76E-07	33	68	0.031964	7.09E-07	43	121	0.058315	7.9E-07
	x3	23	68	0.03217	6.54E-07	23	48	0.026037	6.35E-07	33	99	0.051016	6.65E-07
	x4	29	84	0.035997	5.51E-07	33	68	0.038477	7.09E-07	41	116	0.056119	9.04E-07
	x5	29	84	0.045128	8.8E-07	33	68	0.038126	6.3E-07	40	112	0.055118	8.23E-07
	x6	36	105	0.046939	6.96E-07	33	68	0.032964	9.6E-07	44	124	0.060508	9.1E-07
10,000	x2	32	93	0.248358	6.17E-07	33	68	0.202754	8.65E-07	43	122	0.319696	7.09E-07
	x3	21	63	0.185771	7.63E-07	20	42	0.140302	5.83E-07	33	99	0.288579	6.49E-07
	x4	30	87	0.24375	4.84E-07	33	68	0.195997	8.65E-07	41	117	0.296226	9.89E-07
	x5	30	87	0.226712	5.87E-07	33	68	0.219253	7.71E-07	40	112	0.293811	6.1E-07
	x6	36	105	0.279896	6.21E-07	34	70	0.21272	6.67E-07	44	125	0.321561	8.29E-07
100,000	x2	32	93	2.216175	6.8E-07	34	70	1.824005	6.25E-07	43	123	2.645576	8.51E-07
	x3	21	63	1.485718	7.45E-07	20	42	1.095675	4.7E-07	33	99	2.111431	6.49E-07
	x4	31	90	2.106171	4.43E-07	34	70	1.823136	6.25E-07	43	123	2.684501	6.45E-07
	x5	28	82	1.928036	8.44E-07	33	68	1.745576	9.46E-07	40	113	2.512399	8.23E-07
	x6	36	105	2.461783	7.08E-07	34	70	1.813466	7.95E-07	46	130	2.816934	6.34E-07

Table 4.20: Numerical results of ADLP, IDFDD and ADLCG methods for problem 11

		ADLP				IDFDD				ADLCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
100	x1	28	56	0.091841	7.27E-07	64	128	0.160718	9.9E-07	41	74	0.113752	9.84E-07
	x2	24	49	0.099303	6E-07	49	99	0.171848	9.63E-07	35	62	0.136679	6.56E-07
	x3	28	56	0.112027	6.71E-07	64	128	0.226143	8.19E-07	43	76	0.132411	7.02E-07
	x4	26	53	0.112649	5.67E-07	53	107	0.18895	8.61E-07	37	64	0.112005	8.45E-07
	x5	39	84	0.164482	6.7E-07	74	141	0.238001	8.05E-07	50	92	0.179143	8.42E-07
	x6	24	48	0.108039	8.32E-07	55	109	0.189473	9.83E-07	33	59	0.11219	6.99E-07
1,000	x1	27	54	0.653316	6.95E-07	66	131	1.522788	9.57E-07	42	73	0.901995	8.92E-07
	x2	24	49	0.603825	6E-07	49	99	1.186775	9.63E-07	35	62	0.791811	6.56E-07
	x3	27	54	0.665231	8.23E-07	64	123	1.632213	7.81E-07	42	72	0.91574	6.33E-07
	x4	26	53	0.655574	5.84E-07	53	107	1.295001	8.67E-07	37	64	0.794192	8.18E-07
	x5	38	84	0.943019	6.47E-07	77	146	1.759476	8.56E-07	56	115	1.276215	6.99E-07
	x6	24	48	0.599837	7.33E-07	57	112	1.364328	8.69E-07	38	65	0.854715	7.77E-07
5,000	x1	26	52	12.17649	5.35E-07	69	137	32.89757	7.85E-07	42	74	18.57345	8.49E-07
	x2	24	49	11.41467	6E-07	49	99	23.20703	9.63E-07	35	62	15.56974	6.56E-07
	x3	27	54	12.65706	5.78E-07	69	137	32.44721	9.41E-07	42	73	18.63853	5.52E-07
	x4	26	53	12.34152	5.85E-07	53	107	25.00151	8.68E-07	37	64	16.25942	8.16E-07
	x5	34	71	16.49941	6.88E-07	84	163	38.92182	7.63E-07	61	146	31.54485	9.89E-07
	x6	24	48	11.46317	5.84E-07	60	117	28.08097	9.5E-07	39	70	17.38157	8.97E-07

Table 4.21: Numerical results of ADLP, IDFDD and ADLCG methods for problem 12

		ADLP				IDFDD				ADLCG			
Dimension	SP	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM	NI	FEV	TIME	NORM
100	x1	25	50	0.092739	6.05E-07	50	101	0.167927	7.46E-07	39	71	0.123224	8.33E-07
	x2	24	48	0.096731	8.73E-07	50	101	0.194073	7.38E-07	41	73	0.111879	7.67E-07
	x3	24	48	0.091693	9.97E-07	49	99	0.169799	9.77E-07	40	72	0.103749	7.60E-07
	x4	24	48	0.088172	9.08E-07	49	99	0.203597	9.66E-07	40	72	0.138209	6.80E-07
	x5	28	56	0.08188	6.91E-07	53	107	0.209464	7.60E-07	40	73	0.141121	5.90E-07
	x6	25	50	0.101051	6.37E-07	51	103	0.184793	8.02E-07	42	75	0.147604	5.96E-07
1,000	x1	25	50	0.575409	6.06E-07	53	107	1.199277	8.85E-07	42	75	0.960465	9.44E-07
	x2	24	48	0.567989	8.76E-07	53	107	1.260206	9.05E-07	42	75	0.923941	7.83E-07
	x3	25	50	0.61977	6.13E-07	53	107	1.220053	8.76E-07	41	74	0.912193	7.59E-07
	x4	24	48	0.568745	9.15E-07	53	107	1.238389	8.96E-07	42	75	0.929832	9.78E-07
	x5	28	56	0.684377	9.75E-07	56	113	1.308154	8.53E-07	43	78	0.979088	7.26E-07
	x6	25	50	0.637105	6.75E-07	54	109	1.287852	9.75E-07	45	80	0.964631	5.98E-07
5,000	x1	25	50	11.77355	6.13E-07	56	113	26.57459	7.53E-07	46	81	20.49126	6.11E-07
	x2	24	48	11.42026	8.81E-07	56	113	26.39162	7.73E-07	47	82	20.82066	7.85E-07
	x3	25	50	11.72481	6.22E-07	56	113	26.58983	7.51E-07	46	81	20.40761	7.82E-07
	x4	24	48	11.2997	9.19E-07	56	113	26.50886	7.71E-07	47	82	21.05078	7.82E-07
	x5	29	58	13.74667	5.92E-07	58	117	27.6969	9.95E-07	44	79	19.84518	9.72E-07
	x6	25	50	11.71424	7.17E-07	57	115	26.94505	8.32E-07	46	82	20.46879	7.20E-07

Table 4.22: Summary of test results reported in Table 4.16-4.21

Methods	NI	Percentage	FEV	Percentage	Time	Percentage
ADLP	69	63.89%	60	55.56%	69	63.89%
IDFDD	5	4.63%	15	13.89%	17	15.74%
ADLCG	27	25.00%	27	25.00%	22	20.37%
Undecided	7	6.48%	6	5.55%	0	0.00%

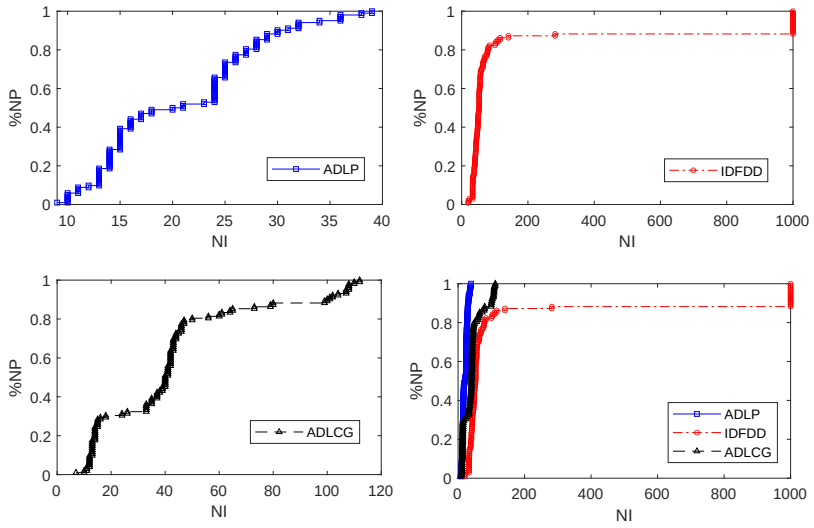

Figure 4.9: Data profile for the number of iterations

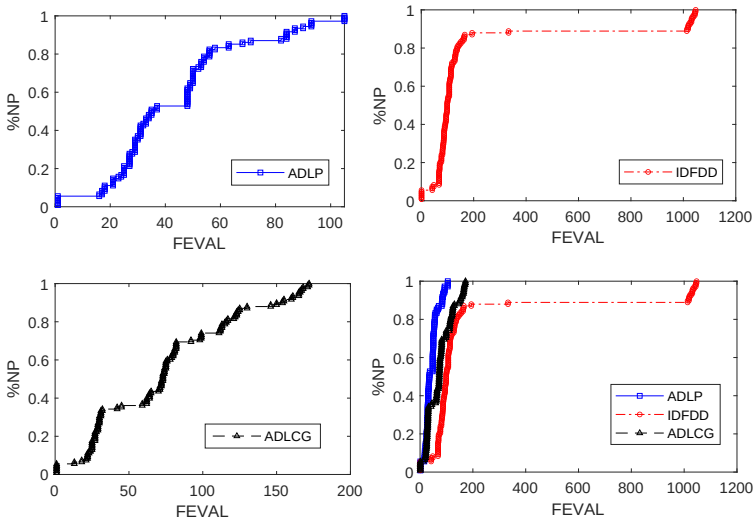

Figure 4.10: Data profile for the functions evaluations

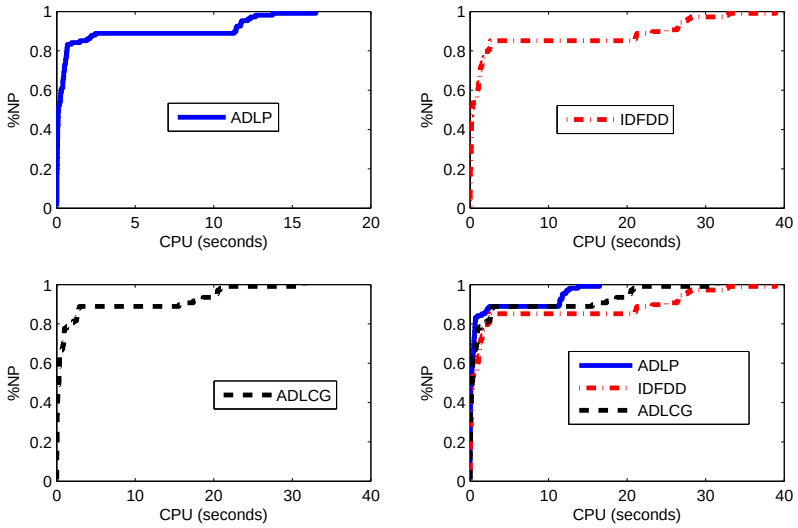

Figure 4.11: Data profile for the CPU time (in second)

Summaries of the results reported in Tables 4.16-4.21 are presented in Tables 4.22 to illustrate which approach is the winner in terms of number of iterations, functions evaluations and CPU time. The summary shows that the ADLP method is the most efficient as it solves 63.89% (69 out of 108) of the problems with least number of iterations than IDFDD and ADLCG methods which solve 4.63% (5 out of 108) and 25.00% (27 out of 108) respectively. Furthermore, the summarized result shows that the two or three methods solve 7 problems with the same number of iteration, which translates to 6.48% and is reported as undecided. The table also shows that the ADLP method solves 63.89% (69 out of 108) of the problems with less CPU time compared to IDFDD and ADLCG methods which solve 15.74% (17 out of 108) and 20.37% (22 out of 108) respectively. Moreover, for the function evaluations, the proposed method solves 55.56% (60 out of 108), IDFDD method solves 13.89% (15 out of 108), and ADLCG method solves 25.00% (27 out of 108), where undecided is 5.55% (6 out of 108).

In order to see the required NI, FEVAL and CPU budget for each of the three methods to successfully solve the test problems in this experiment, we plot data profile for NI, FEVAL and CPU presented in Tables 4.16–4.21 using the the data profile of Mor*é* and Stefan [80]. Figure 4 plots %NP (percentage of the number of problem) versus NI while Figure 5 plots %NP versus FEVAL and Figure 6 plots %NP versus CPU. Figure 4.9-4.11 show the required NI, FEVAL and CPU budget for each method to solve certain percentage of the 108 problems considered for this experiment. It can seen from Figure 4.9 (top-left, top-right and bottom left) that with the budget of 40 NI, the ADLP method successfully solve all the test problems compare to IDFDD and ADLCG solves less than 40% of the test problems with the same budget. Also, Figure4.10 (top-left, top-right and bottom left) shows that with the budget of 100 FEVAL, the ADLP method successfully solve the entire test problems while IDFDD and ADLCG solve 10% and 75% of the problems, respectively. From Figure 4.11 (top-left, top-right and bottom left) that with the budget of 20 CPU time (in second), the ADLP method successfully solve all the test problems compare to IDFDD and ADLCG that repectively solve than 85% and 90%of the test problems with the same budget. This means that IDFDD and ADLCG methods are more computationally expensive compared to

ADLP method. Taking the overall performance into consideration, it can be concluded that the ADLP method competes favourably with the existing methods and is reasonably less computational expensive.

4.6 Conclusion

In this chapter, an accelerated Dai-Liao projection (ADLP) method for convex constrained monotone nonlinear equations is presented. A new value for the Dai-Liao parameter $t > 0$ was successfully derived using the Newton approach. The proposed method has been proved to be globally convergent under some standard assumptions. Furthermore, the proposed method has been extended to solve general system of nonlinear equations. Numerical comparisons were made on some large-scale test problems which demonstrated the efficacy of the proposed method. In addition, Tables 4.9-4.21 and Figures 4.6-4.11 have shown that the proposed method is practically welcome, as it has the least number of iterations and CPU times than the compared methods.

CHAPTER FIVE

Image Deblurring Problems through Accelerated Hager-Zhang Projection Method for Convex Constrained Monotone Nonlinear Equations

5.1 Introduction

Many researchers have developed interest in finding new choices for the Hager-Zhang nonnegative parameter and developed schemes that generate descent search directions. In this chapter, a CG method to solve constrained monotone nonlinear equations (1.3) is presented.

Some iterative approaches for solving these problems include derivative-free methods [60, 61, 111, 4], double step length methods [58, 63, 64], double direction methods [62, 59], Newton methods and their improved version, i.e., the quasi-Newton methods [43, 108, 125, 69]. Typically, iterative procedure of the aforementioned methods are established by

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, \ldots,$$

where, $s_k = x_{k+1} - x_k$, $\alpha_k > 0$ is a step length that can be computed using any suitable line search, and the search directions d_k can be computed as

$$d_k = -F_k + R_k. \tag{5.137}$$

It is easy to observe that if $R_k = 0$, then (5.137) reduces to the steepest decent direction. However, the Newton direction can be obtained if $R_k = (I - (F_k')^{-1})F_k$,

where I is an identity matrix and F_k' is the Jacobian matrix. Consequently, we can find the quasi-Newton directions whenever $R_k = (I - B_k^{-1})F_k$, where B_k is $n \times n$ matrix that approximate the Jacobian matrix.

Regardless of the attractive features of the Newton and quasi-Newton methods, the Jacobian matrix or its approximation is calculated at each iteration. This renders the schemes not ideal for solving large-scale problems because huge matrix storage is required at each iteration, which is costly in numerical experiments. To overcome these shortfall, matrix-free methods are proposed. Barzilai and Borwein [20] proposed one of the successful matrix-free methods that generates a sequence of iterates as $x_{k+1} = x_k - \tau_k F_k$ and τ_k is the step length, which can be written as $x_{k+1} = x_k - D_k F_k$, where $D_k = \tau_k I$ which is supposed to satisfy the secant equation (2.30). Another variant of matrix-free approaches are the conjugate gradient (CG) methods. One can easily observe that if $d_0 = -F_0$, and $R_k = \beta_k d_{k-1}$, then (5.137) reduces to classical conjugate gradient directions (1.12).

The proposed method is based on presenting a new value of the Hager-Zhang parameter θ. This is achieved by combining the conjugate gradient method with the Newton method approach. Moreover, to solve the large-scale problems, the Jacobian matrix is approximated via acceleration parameter. Under some mild conditions, the proposed method is proven to be globally convergent and numerical experiments conducted show the efficacy of the proposed method. In addition, the proposed method is successfully applied to handle the ℓ_1-norm regularization problem in image recovery, which exhibits a better result than the existing method in the previous literature. Throughout this chapter, the space $\mathbb{R}^n$ denote the $n-$dimensional real space equipped with the Euclidean norm $\|\cdot\|$, $F_k = F(x_k)$, and $F_k' = F'(x_k)$.

5.2 New choice of Hager-Zhang nonnegative parameter

Consider the Hager-Zhang CG parameter defined in [112] as

$$\beta_k^{HZ}(\theta) = \frac{F_k^T y_{k-1}}{s_{k-1}^T y_{k-1}} - \theta_k \frac{\|y_{k-1}\|^2 F_k^T s_{k-1}}{(s_{k-1}^T y_{k-1})^2}, \tag{5.138}$$

where, the Hager-Zhang nonnegative parameter θ_k is given as

$$\theta_k = P - Q\frac{(s_{k-1}^T y_{k-1})^2}{\|s_{k-1}\|^2\|y_{k-1}\|^2}, \quad P \geq \frac{1}{4}, \quad Q \leq 0.$$

For general function F, the CG parameter $\beta_k^{HZ}(\theta)$ at some iteration may be undefined when its denominator becomes zero. For this reason, we propose the following modified Hager-Zhang CG parameter

$$\beta_k^{AHZP}(\theta) = \frac{F_k^T w_{k-1}}{s_{k-1}^T \psi_{k-1}} - \theta_k \frac{\|w_{k-1}\|^2 F_k^T s_{k-1}}{(s_{k-1}^T \psi_{k-1})^2}. \tag{5.139}$$

Here,

$$\psi_{k-1} = w_{k-1} + \left(1 + \max\left\{0, -\frac{s_{k-1}^T w_{k-1}}{\|s_{k-1}\|^2}\right\}\right) s_{k-1}, \quad w_{k-1} = F_k - F_{k-1} + r s_{k-1},\ r > 0,$$

and $s_{k-1} = z_k - x_k = \alpha_k d_k$. Our search direction is therefore defined as follows.

$$d_k = \begin{cases} -F_k, & \text{if} \quad k = 0, \\ -\eta_k F_k + \beta_k^{AHZP}(\theta) s_{k-1}, & \text{if} \quad k \geq 1, \end{cases} \tag{5.140}$$

where η_k is a positive parameter that can be chosen so that search direction d_k satisfies the following property

$$F_k^T d_k \leq -c\|F_k\|^2, \quad c > 0. \tag{5.141}$$

Remark 5.2.1 *We give the following remark.*
If $\max\left\{0, -\frac{s_{k-1}^T w_{k-1}}{\|s_{k-1}\|^2}\right\} \neq 0$ *in the definition of* ψ_{k-1}*, then*

$$\begin{aligned} s_{k-1}^T \psi_{k-1} &= s_{k-1}^T \left(w_{k-1} + s_{k-1} - \frac{s_{k-1}^T w_{k-1}}{\|s_{k-1}\|^2} s_{k-1} \right) \\ &= s_{k-1}^T w_{k-1} + \|s_{k-1}\|^2 - \frac{s_{k-1}^T w_{k-1}}{\|s_{k-1}\|^2} \|s_{k-1}\|^2 \\ &= \|s_{k-1}\|^2 > 0. \end{aligned} \tag{5.142}$$

Otherwise,

$$\begin{aligned} s_{k-1}^T \psi_{k-1} &= s_{k-1}^T w_{k-1} + \|s_{k-1}\|^2 \\ &= s_{k-1}^T (F_k - F_{k-1} + r s_{k-1}) + \|s_{k-1}\|^2 \\ &= s_{k-1}^T (F_k - F_{k-1}) + r\|s_{k-1}\|^2 + \|s_{k-1}\|^2 \\ &\geq (r+1)\|s_{k-1}\|^2 > 0. \end{aligned} \tag{5.143}$$

Since the function F *is monotone, the last inequality follows. This shows that* $s_{k-1}^T \psi_{k-1} > 0$ *provided that the solution is not reached. Therefore,* $\beta_k^{AHZP}(\theta)$ *is well-defined.*

On the other hand, if a point x_k is sufficiently close to the solution say x^*, then Newton's direction is the one to follow. Therefore, from the CG direction in (5.140) and the Hager-Zhang CG parameter in (5.139), the value of parameter θ_k can be computed as follows:

$$-(F_k')^{-1} F_k = -F_k + \frac{F_k^T w_{k-1}}{s_{k-1}^T \psi_{k-1}} s_{k-1} - \theta_k \frac{\|w_{k-1}\|^2 F_k^T s_{k-1}}{(s_{k-1}^T \psi_{k-1})^2} s_{k-1}. \tag{5.144}$$

With a view to avoiding the computation of the Jacobian matrix F_k' and its inverse, we are interested in approximating the Jacobian matrix via

$$F_k' \approx \gamma_k I, \tag{5.145}$$

where, γ_k is an acceleration parameter presented as

$$\gamma_k = \tau^{BB1}(\tau^{BB2})^{-1} = \frac{\|y_{k-1}\|^2\|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2}, \tag{5.146}$$

where, τ^{BB1} and τ^{BB2} are in (2.15). However, (5.145) can be modified as

$$F_k' \approx \hat{\gamma}_k I, \tag{5.147}$$

with γ_k in (5.146) modified as

$$\hat{\gamma}_k = \frac{\|w_{k-1}\|^2\|s_{k-1}\|^2}{(\psi_{k-1}^T s_{k-1})^2}. \tag{5.148}$$

By substituting (5.147) and (5.148) in (5.144), and applying some algebraic calculation, the proposed value of the parameter θ_k can be obtained as

$$\theta_k = \frac{F_k^T s_{k-1} - \frac{F_k^T s_{k-1}\|w_{k-1}\|^2\|s_{k-1}\|^2}{(\psi_{k-1}^T s_{k-1})^2} + \frac{F_k^T w_{k-1}\|w_{k-1}\|^2\|s_{k-1}\|^4}{(\psi_{k-1}^T s_{k-1})^3}}{\frac{F_k^T s_{k-1}\|w_{k-1}\|^4\|s_{k-1}\|^4}{(\psi_{k-1}^T s_{k-1})^4}}. \tag{5.149}$$

To ensure that the proposed θ_k is nonnegative, we incorporate to the idea presented in [7]. Therefore the value of the proposed nonnegative parameter denoted as $\hat{\theta}_k$ can be presented as

$$\hat{\theta}_k = \max\left\{\theta_k, \tau\frac{\|w_{k-1}\|^2}{\psi_{k-1}^T s_{k-1}}\right\}. \tag{5.150}$$

where, $\tau \geq \frac{1}{4}$.

Remark 5.2.2 *Now, we give the following remark.*
For $k = 0$, we have $F_0^T d_0 = -\|F_0\|^2$. This satisfies (7.227) with $c = 1$.

For $k \geq 0$, we have

$$
\begin{aligned}
F_k^T d_k &= F_k^T \left(-\eta_k F_k + \frac{F_k^T w_{k-1}}{s_{k-1}^T \psi_{k-1}} s_{k-1} - \hat{\theta}_k \frac{\|w_{k-1}\|^2 F_k^T s_{k-1}}{(s_{k-1}^T \psi_{k-1})^2} s_{k-1} \right) \\
&= -\eta_k \|F_k\|^2 + \frac{(F_k^T w_{k-1})(F_k^T s_{k-1})}{s_{k-1}^T \psi_{k-1}} - \hat{\theta}_k \frac{\|w_{k-1}\|^2 (F_k^T s_{k-1})^2}{(s_{k-1}^T \psi_{k-1})^2} \\
&\leq -\eta_k \|F_k\|^2 + \frac{\|F_k\|^2 \|w_{k-1}\| \|s_{k-1}\|}{s_{k-1}^T \psi_{k-1}} \\
&\leq -\eta_k \|F_k\|^2 + \frac{\|F_k\|^2 \|w_{k-1}\| \|s_{k-1}\|}{\|s_{k-1}\|^2} \\
&\leq -\eta_k \|F_k\|^2 + \frac{\|F_k\|^2 \|w_{k-1}\|}{\|s_{k-1}\|} \\
&\leq -\left(\eta_k - \frac{\|w_{k-1}\|}{\|s_{k-1}\|} \right) \|F_k\|^2. \qquad (5.151)
\end{aligned}
$$

For the search direction (5.140) to satisfy (7.227), we need

$$
\eta_k \geq c + \frac{\|w_{k-1}\|}{\|s_{k-1}\|}.
$$

Without the loss of generality, we select

$$
\eta_k = c + \frac{\|w_{k-1}\|}{\|s_{k-1}\|}. \qquad (5.152)
$$

Algorithm 4: Accelerated Hager-Zhang projection Method(AHZP)

Input: Given $x_0 \in \Omega$, $d_0 = -F_0$, $\rho \in (0,1)$, $\sigma > 0$, $\varepsilon = 10^{-6}$, $0 < \zeta < 2$, and $\xi > 0$, set $k = 0$.

Step 1: Compute F_k.

Step 2: If $\|F_k\| \leq \varepsilon$ then stop, else go to Step 3.

Step 3: Let $\alpha_k = \xi \rho^{m_k}$, with m_k being the smallest nonnegative integer m such that

$$-F(x_k + \alpha_k d_k)^T d_k \geq \sigma \alpha_k \|F(x_k + \alpha_k d_k)\| \|d_k\|^2. \tag{5.153}$$

Step 4: Set $z_k = x_k + \alpha_k d_k$.

Step 5: If $z_k \in \Omega$ and $\|F(z_k)\| \leq \varepsilon$, stop, otherwise go to Step 6.

Step 6: Compute the next iterate by

$$x_{k+1} = P_\Omega[x_k - \zeta \lambda_k F(z_k)], \tag{5.154}$$

where, $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step 7: Compute the search direction

$$d_{k+1} = -\eta_{k+1} F_{k+1} + \left(\frac{F_{k+1}^T w_k}{s_k^T \psi_k} - \hat{\theta}_{k+1} \frac{\|w_k\|^2 F_{k+1}^T s_k}{(s_k^T \psi_k)^2} \right) s_k.$$

where $\hat{\theta}_{k+1}$ is defined in (5.150).

Step 8: Consider $k = k+1$ and go to Step 2.

Remark 5.2.3 *We give the following remark.*

(a) From assumption (A1) we have

$$\|w_{k-1}\| = \|F(z_{k-1}) - F(x_{k-1}) + r s_{k-1}\| \leq \|F(z_{k-1}) - F(x_{k-1})\| + r\|s_{k-1}\| \leq (L+r)\|s_{k-1}\|. \tag{5.155}$$

(b) To establish the global convergence property of Algorithm 4, the proposed $\hat{\theta}_k$ *in (5.150) needs to be bounded. Therefore, we set*

$$\hat{\theta}_k \leftarrow \min\{\hat{\theta}_k, H\}, \tag{5.156}$$

where H *is a suitable positive number.*

5.3 Convergence Analysis of Algorithm 4 (AHZP)

The convergence analysis of Algorithm 4 is established in this subsection.

Lemma 5.3.1 *Suppose assumptions (A1)-(A3) hold and let* $\{x_k\}$ *and* $\{z_k\}$ *be generated by Algorithm 4, then* $\{x_k\}$ *and* $\{z_k\}$ *are bounded. In addition, we have*

$$\|d_k\| \leq M, \tag{5.157}$$

$$\lim_{k\to\infty} \|z_k - x_k\| = 0, \tag{5.158}$$

$$\lim_{k\to\infty} \|x_k - x_{k+1}\| = 0. \tag{5.159}$$

Proof To show the boundedness of the sequences $\{x_k\}$ and $\{z_k\}$, let $\bar{x} \in S^x$ be any solution of (1.3). Then by monotonicity of F we can write

$$(x_k - \bar{x})^T F(z_k) \geq (x_k - z_k)^T F(z_k). \tag{5.160}$$

Using the line search condition (5.153) and definition of z_k, we have

$$(x_k - z_k)^T F(z_k) \geq \sigma \alpha_k^2 \|F(z_k)\| \|d_k\|^2 > 0. \tag{5.161}$$

Also, using (3.48) and fact that $x_{k+1} = P_\Omega[x_k - \zeta\lambda_k F(z_k)]$, we have

$$\begin{aligned}
\|x_{k+1} - \bar{x}\|^2 &= \|P_\Omega(x_k - \zeta\lambda_k F(z_k)) - \bar{x}\|^2 \\
&\leq \|x_k - \zeta\lambda_k F(z_k) - \bar{x}\|^2 \\
&= \|x_k - \bar{x}\|^2 - 2\zeta\lambda_k (x_k - \bar{x})^T F(z_k) + \zeta^2\lambda_k^2 \|F(z_k)\|^2 \\
&\leq \|x_k - \bar{x}\|^2 - 2\zeta\lambda_k (x_k - z_k)^T F(z_k) + \zeta^2\lambda_k^2 \|F(z_k)\|^2 \\
&= \|x_k - \bar{x}\|^2 - \zeta(2-\zeta)\frac{((x_k - z_k)^T F(z_k))^2}{\|F(z_k)\|^2} \qquad (5.162) \\
&\leq \|x_k - \bar{x}\|^2 - \widehat{\sigma}\|x_k - z_k\|^4. \qquad (5.163)
\end{aligned}$$

where $\widehat{\sigma} = \zeta(2-\zeta)\sigma^2 > 0$.
Since $0 < \zeta < 2$, from (5.162) we obtain

$$\|x_{k+1} - \bar{x}\| \leq \|x_k - \bar{x}\|. \qquad (5.164)$$

This recursively implies that $\|x_k - \bar{x}\| \leq \|x_0 - \bar{x}\|$, for all k. So, $\{\|x_k - \bar{x}\|\}$ is clearly a decreasing sequence, which implies that $\{x_k\}$ is bounded. Furthermore, utilizing assumption (A1), (A3) and (5.164) we have

$$\|F(x_k)\| = \|F(x_k) - F(\bar{x})\| \leq L\|x_k - \bar{x}\| \leq L\|x_0 - \bar{x}\|. \qquad (5.165)$$

Let $m_1 = L\|x_0 - \bar{x}\|$, then we have

$$\|F(x_k)\| \leq m_1. \qquad (5.166)$$

From (5.140), (5.142), (5.151),(9.276), (5.155), and(5.156) we have

$$
\begin{aligned}
\|d_k\| &= \| -\eta_k F_k + \beta_k^{AHZP}(\theta) s_{k-1}\| \\
&\leq \left| c + \frac{\|w_{k-1}\|}{\|s_{k-1}\|} \right| \|F_k\| + \left| \frac{F_k^T w_{k-1}}{s_{k-1}^T \psi_{k-1}} - \hat{\theta}_k \frac{\|w_{k-1}\|^2 F_k^T s_{k-1}}{(s_{k-1}^T \psi_{k-1})^2} \right| \|s_{k-1}\| \\
&\leq \left[c + \frac{\|w_{k-1}\|}{\|s_{k-1}\|} \right] \|F_k\| + \left| \frac{F_k^T w_{k-1}}{s_{k-1}^T \psi_{k-1}} \right| \|s_{k-1}\| + \left| \hat{\theta}_k \frac{\|w_{k-1}\|^2 F_k^T s_{k-1}}{(s_{k-1}^T \psi_{k-1})^2} \right| \|s_{k-1}\| \\
&\leq c\|F_k\| + \frac{\|w_{k-1}\|}{\|s_{k-1}\|} \|F_k\| + \frac{\|w_{k-1}\| \|s_{k-1}\|}{s_{k-1}^T \psi_{k-1}} \|F_k\| + \hat{\theta}_k \frac{\|w_{k-1}\|^2 \|s_{k-1}\|^2}{(s_{k-1}^T \psi_{k-1})^2} \|F_k\| \\
&\leq c\|F_k\| + \frac{(L+r)\|s_{k-1}\|}{\|s_{k-1}\|} \|F_k\| + \frac{(L+r)\|s_{k-1}\|^2}{\|s_{k-1}\|^2} \|F_k\| + H \frac{(L+r)^2 \|s_{k-1}\|^4}{\|s_{k-1}\|^4} \|F_k\| \\
&= [c + 2(L+r) + H(L+r)^2] \|F_k\| \\
&\leq [c + 2(L+r) + H(L+r)^2] m_1. \qquad (5.167)
\end{aligned}
$$

Where Cauchy-Schwarz inequality is applied in the third inequality. The fourth inequality follow from Remark (5.2.1) and Remark (5.2.3) respectively.
Taking $M = [c + 2(L+r) + H(L+r)^2] m_1$ we have (5.157).

Next, since the sequences $\{x_k\}$ and $\{d_k\}$ are bounded, then the definition z_k in Step 3 of Algorithm 4, it holds that $\{z_k\}$ is also bounded. Therefore, similar argument as in (5.165), there exists some constants, say $\kappa > 0$, such that

$$
\|F(z_k)\| \leq \kappa. \qquad (5.168)
$$

Now, from (5.162), we have

$$
((x_k - z_k)^T F(z_k))^2 \leq \frac{\|F(z_k)\|^2}{\zeta(2-\zeta)} (\|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2). \qquad (5.169)
$$

From the line search (5.153), we have

$$
\alpha_k^4 \|d_k\|^4 \leq \frac{\alpha_k^2}{\sigma^2} (F(z_k)^T d_k)^2. \qquad (5.170)
$$

Combining (5.162) and (5.170), it holds

$$\alpha_k^4\|d_k\|^4 \leq \frac{\|F(z_k)\|^2}{\sigma^2\zeta(2-\zeta)}(\|x_k-\bar{x}\|^2-\|x_{k+1}-\bar{x}\|^2). \tag{5.171}$$

Since the sequence $\{\|x_k-\bar{x}\|\}$ is convergent, using the fact $\sigma>0$, $0<\zeta<2$ and $\{F(z_k)\}$ is bounded, taking limit on both sides of (5.171) yields

$$\lim_{k\to\infty}\alpha_k^4\|d_k\|^4 \leq 0,$$

and hence it holds that

$$\lim_{k\to\infty}\alpha_k\|d_k\| = 0. \tag{5.172}$$

Combining (5.171) with the definition of z_k implies (5.158) holds. On the other hand, from (3.48) and the definition of λ_k we have

$$\begin{aligned}\|x_{k+1}-x_k\| &= \|P_\Omega(x_k-\zeta\lambda_k F(z_k))-x_k\| \\ &\leq \|x_k-\zeta\lambda_k F(z_k)-x_k\| \\ &= \|\zeta\lambda_k F(z_k)\| \\ &\leq \zeta\|x_k-z_k\|.\end{aligned} \tag{5.173}$$

This implies (5.159). ■

Theorem 5.3.2 *Suppose assumption (A1)-(A3) hold and $\{x_k\}$ be generated by Algorithm 4. Then*

$$\liminf_{k\to\infty}\|F(x_k)\| = 0. \tag{5.174}$$

Proof For the sake of contradiction, suppose that (5.174) is not true, then there exists $\delta_0>0$ such that

$$\|F(x_k)\| \geq \delta_0 \quad \text{holds}, \quad \forall k>0. \tag{5.175}$$

Suppose that the search direction $\|d_k\|\neq 0$ unless at the solution, then there is a constant, say δ_1

$$\|d_k\| \geq \delta_1. \tag{5.176}$$

If $\alpha_k \neq \xi$, then by the definition of α_k, $\rho^{-1}\alpha_k$ does not satisfy (5.153) i.e.,

$$-F(x_k+\rho^{-1}\alpha_k d_k)^T d_k < \sigma\rho^{-1}\alpha_k \|F(z_k)\|\|d_k\|^2.$$

Now, combining with (5.151) and (9.276) gives

$$\begin{aligned} c\|F(x_k)\|^2 &\leq -F(x_k)^T d_k \\ &= (F(x_k+\rho^{-1}\alpha_k d_k)-F(x_k))^T d_k - F(x_k+\rho^{-1}\alpha_k d_k)^T d_k \\ &\leq L\rho^{-1}\alpha_k\|d_k\|^2 + \sigma\rho^{-1}\alpha_k\|F(z_k)\|\|d_k\|^2 \\ &= \alpha_k\|d_k\|(L+\sigma\kappa)\rho^{-1}\|d_k\|. \end{aligned} \tag{5.177}$$

Therefore, from (5.177), we have

$$\alpha_k\|d_k\| > \frac{\rho}{(L+\sigma\kappa)}\frac{c\|F_k\|^2}{\|d_k\|} \geq \frac{\rho}{(L+\sigma\kappa)}\frac{c\delta_0^2}{M}. \tag{5.178}$$

The inequality (5.178) contradicts (5.172). Therefore (5.174) holds. Hence, the proof is completed. ■

5.4 Numerical Experiments

In this section, numerical results are presented to demonstrate the efficacy of the AHZP method by comparing it to the following existing CG methods.

- DLPM: Algorithm 1 proposed in [1] with
  $\theta_k = p\frac{\|y_{k-1}\|^2}{y_{k-1}^T s_{k-1}} - q\frac{y_{k-1}^T s_{k-1}}{\|s_{k-1}\|^2}$, $p \geq \frac{1}{4}$, $q \leq 0$.
- MHZ2: Algorithm 2 proposed in [93] with

$$\theta_k = \frac{\|s_{k-1}\|^2\|\hat{y}_{k-1}\|^2}{(s_{k-1}^T\hat{y}_{k-1})^2} + \sqrt{\frac{(s_{k-1}^T\hat{y}_{k-1})^2}{\|s_{k-1}\|^2\|\hat{y}_{k-1}\|^2} + \frac{\|s_{k-1}\|^4\|\hat{y}_{k-1}\|^4}{(s_{k-1}^T\hat{y}_{k-1})^4}},$$

$$\hat{y}_{k-1} = y_{k-1} + \rho\frac{\max\{v_{k-1},0\}s_{k-1}}{\|s_{k-1}\|}, \quad \rho = 0.1,$$

$$v_{k-1} = 6(\|F_{k-1}\| - \|F_k\|) + 3(F_{k-1}+F_k)^T s_{k-1}.$$

The codes used are written in Matlab 8.3.0 (R2014a) and run on a personal computer equipped with a 1.80 GHz CPU processor and 8 GB RAM. When implementing our algorithm (AHZP) in this experiments, the following parameters are set; $\xi = 1$, $\sigma = 10^{-4}$, $\rho = 0.9$, $\tau = 0.4$, and $\zeta = 1.3$. However, the parameters of DLPM and MHZ2 algorithm, are taken as in [2] and [93] respectively. The iteration is set to stop for all the three methods if the following conditions occur: (i) when $\|F_k\| \leq 10^{-7}$, (ii) when $\|F(z_k)\| \leq 10^{-7}$, or (iii) when the iterations exceed 1000 but no x_k satisfying the stopping criterion is obtained. We have carried out the numerical experiments of the three methods on the previous seven test problems with different initial points with dimension (n values) 1000, 10,000 and 100,000. The numerical results of the three (3) methods are reported in Tables 5.23-5.26. From the Tables,

- IP represents the starting points,
- Iter represents the total number of iterations,
- Time (s) represents the CPU time (in seconds),
- Funeval represents the function evaluations,
- $\|F_k\|$ is the norm of the residual at the termination point.

The initial points used in the experiment are as follows: $a = (1,1,...,1)^T$, $b = \left(\frac{3}{5},\frac{3}{5},...,\frac{3}{5}\right)^T$, $c = \left(\frac{1}{2},\frac{1}{2},...,\frac{1}{2}\right)^T$, $d = \left(\frac{2}{5},\frac{2}{5},...,\frac{2}{5}\right)^T$, $e = \left(\frac{1}{10},\frac{1}{10},...,\frac{1}{10}\right)^T$, $f = \left(1,\frac{1}{2},\frac{1}{3},...,\frac{1}{n}\right)^T$, $g = \left(\frac{1}{4},\frac{-1}{4},...,\frac{(-1)^n}{4}\right)^T$, $h = \left(-\frac{1}{2},-\frac{1}{2},...,-\frac{1}{2}\right)^T$, $i = \left(\frac{1}{2},\frac{1}{2^2},...,\frac{1}{2^n}\right)^T$, and $j = rand(0,1)$ which is randomly selected numbers between 0 and 1.

The following test problems were used in the experiments:

Problem 1 [93]

$F_1(x) = e^{x_1} - 1,$

$F_i(x) = e^{x_i} + x_{i-1} - 1, \quad i = 2,3,\ldots,n-1,$

where $\Omega = \mathbb{R}^n_+$.

Problem 2 [1]

$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 3

$F_i(x) = \cos(x_i) + x_i - 1, \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 4 [1]

$F_i(x) = e^{x_i} - 1, \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 5 [13]

$\frac{i}{n}F_i(x) = e^{x_i} - 1, \quad i = 1,2,\ldots,n,$

where $\Omega = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i > -1, \quad i = 1,2,\ldots,n\}$.

Problem 6 [12]

$F_i(x) = 2x_i - \sin|x_i - 1|, \quad i = 1,2,\ldots,n,$

where $\Omega = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i > -1, \quad i = 1,2,\ldots,n\}$.

Problem 7 [13]

$F_i(x) = e^{x_i^2} + \frac{3}{2}\sin(2x_i) - 1 \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Table 5.23: Numerical results of AHZP, DLPM and MHZ2 methods for problems 1 & 2

			AHZP				DLPM				MHZ2			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
1	1000	a	2	3	0.021529	0	18	20	0.024722	4.69E-08	38	78	0.101556	5.89E-08
		b	2	3	0.002731	0	18	20	0.01409	3.65E-08	44	90	0.111216	7.19E-08
		c	2	3	0.006608	0	18	20	0.016233	3.56E-08	7	16	0.029903	2.32E-08
		d	2	3	0.006045	0	17	19	0.011218	9.27E-08	42	86	0.102123	8.81E-08
		e	2	3	0.004337	0	16	18	0.019832	9.07E-08	13	28	0.038283	4.00E-09
		f	10	11	0.018082	9.44E-08	21	23	0.020321	5.21E-08	19	40	0.061437	6.38E-09
		g	9	10	0.017397	8.86E-09	1	2	0.002652	0	8	18	0.013331	1.30E-08
		h	9	10	0.014801	5.45E-08	1	2	0.002754	0	8	18	0.020729	5.65E-08
		i	9	10	0.011645	8.60E-08	21	22	0.018334	9.24E-08	18	38	0.068956	1.31E-08
		j	9	10	0.014222	1.61E-08	18	20	0.015947	9.03E-08	23	47	0.075136	4.71E-09
	10,000	a	2	3	0.017454	0	18	20	0.121666	4.31E-08	11	24	0.213339	4.36E-09
		b	2	3	0.018829	0	19	21	0.107768	4.01E-08	9	20	0.159113	2.02E-09
		c	2	3	0.019676	0	19	21	0.104575	3.94E-08	9	20	0.187519	1.31E-09
		d	2	3	0.026379	0	19	21	0.100954	3.61E-08	11	24	0.191643	1.16E-08
		e	2	3	0.019183	0	18	20	0.095392	3.53E-08	28	58	0.641475	9.68E-08
		f	10	11	0.086672	5.17E-08	23	25	0.129757	3.98E-08	21	44	0.526154	5.21E-08
		g	2	3	0.016616	0	1	2	0.007503	0	6	14	0.053974	6.88E-08
		h	2	3	0.026333	0	1	2	0.014482	0	8	18	0.087394	8.85E-08
		i	9	10	0.064302	8.60E-08	21	22	0.110167	9.24E-08	18	38	0.401052	1.31E-08
		j	9	10	0.083431	4.92E-08	20	21	0.110579	9.63E-08	25	52	0.593106	6.38E-08
	100,000	a	2	3	0.195079	0	23	25	0.898602	4.57E-08	56	114	6.564013	9.81E-08
		b	2	3	0.488524	0	21	23	0.838763	4.44E-08	31	64	4.241261	9.05E-09
		c	2	3	0.159751	0	21	23	0.795528	3.95E-08	59	120	9.130102	8.01E-08
		d	2	3	0.194665	0	20	22	0.764689	9.58E-08	14	30	2.122003	1.37E-08
		e	2	3	0.153238	0	19	21	0.733677	3.93E-08	16	34	3.601104	2.13E-09
		f	10	11	0.956477	4.73E-08	24	26	1.004896	6.23E-08	12	26	1.890105	4.72E-08
		g	2	3	0.157743	0	1	2	0.050328	0	6	14	0.421217	2.55E-08
		h	2	3	0.114538	0	1	2	0.059275	0	7	16	0.485018	4.16E-08
		i	9	10	0.492416	8.60E-08	21	22	0.780008	9.24E-08	18	38	3.253116	1.31E-08
		j	10	11	0.618087	1.48E-08	21	23	0.815919	3.69E-08	31	63	5.474016	7.47E-09
2	1000	a	2	3	0.003886	0	8	10	0.006623	8.29E-08	7	16	0.012408	3.38E-08
		b	2	3	0.003534	0	8	10	0.005375	6.42E-08	7	16	0.008799	1.56E-08
		c	2	3	0.005533	0	8	10	0.008875	5.62E-08	7	16	0.008466	1.74E-08
		d	2	3	0.004285	0	8	10	0.005965	4.69E-08	7	16	0.012986	1.68E-08
		e	2	3	0.006192	0	8	10	0.008789	1.26E-08	6	14	0.009646	8.43E-08
		f	17	19	0.021996	3.43E-08	13	14	0.010878	5.98E-08	8	18	0.012509	3.03E-08
		g	1	2	0.003083	0	1	2	0.004217	0	1	3	0.003718	0
		h	1	2	0.004019	0	1	2	0.004046	0	1	3	0.003747	0
		i	2	3	0.004093	0	9	11	0.008247	9.77E-08	7	16	0.009015	4.24E-08
		j	15	17	0.016228	4.32E-08	15	17	0.023813	4.84E-08	11	24	0.014605	2.12E-08
	10,000	a	2	3	0.016626	0	9	11	0.036462	2.62E-08	8	18	0.067841	0
		b	2	3	0.014115	0	9	11	0.03275	2.03E-08	7	16	0.064425	4.92E-08
		c	2	3	0.015007	0	9	11	0.047928	1.78E-08	7	16	0.058328	5.50E-08
		d	2	3	0.01523	0	9	11	0.036385	1.48E-08	7	16	0.058226	5.32E-08
		e	2	3	0.014412	0	8	10	0.034329	3.98E-08	7	16	0.057968	1.72E-08
		f	17	19	0.119694	3.42E-08	15	17	0.072436	3.56E-08	8	18	0.065255	2.95E-08
		g	1	2	0.009407	0	1	2	0.011902	0	1	3	0.016767	0
		h	1	2	0.007121	0	1	2	0.009699	0	1	3	0.016486	0
		i	2	3	0.015029	0	9	11	0.034332	9.77E-08	7	16	0.068562	4.24E-08
		j	17	19	0.112661	7.01E-08	16	17	0.072877	3.23E-08	11	24	0.106304	6.47E-08
	100,000	a	2	3	0.097043	0	11	13	0.356388	6.27E-08	8	18	0.459701	0
		b	2	3	0.103969	0	10	12	0.270832	5.04E-08	8	18	0.458115	0
		c	2	3	0.094429	0	9	11	0.252986	5.62E-08	8	18	0.455731	0
		d	2	3	0.102661	0	9	11	0.236187	4.69E-08	8	18	0.460901	0
		e	2	3	0.102626	0	9	11	0.241462	1.26E-08	7	16	0.447212	5.45E-08
		f	17	19	0.901923	3.42E-08	13	15	0.434628	8.61E-08	8	18	0.489511	2.95E-08
		g	1	2	0.060183	0	1	2	0.057159	0	1	3	0.122991	0
		h	1	2	0.056239	0	1	2	0.069409	0	1	3	0.126692	0
		i	2	3	0.094707	0	9	11	0.270585	9.77E-08	7	16	0.450833	4.24E-08
		j	19	21	0.975142	4.74E-08	16	17	0.446619	7.62E-08	12	26	0.694912	0

Table 5.24: Numerical results of AHZP, DLPM and MHZ2 methods for problems 3 & 4

			AHZP				DLPM				MHZ2			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
3	1000	a	3	4	0.006845	0	7	9	0.010683	4.21E-08	4	10	0.007088	1.52E-12
		b	2	3	0.003896	0	6	8	0.010063	4.6E-08	4	10	0.008786	1.07E-10
		c	2	3	0.003372	0	1	2	0.004354	0	4	10	0.007072	1.29E-12
		d	1	2	0.002406	0	6	8	0.014063	4.88E-08	3	8	0.007188	1.97E-08
		e	1	2	0.002909	0	5	7	0.011653	4.86E-08	4	10	0.007445	0
		f	17	18	0.022451	6.73E-08	15	17	0.025455	8.31E-08	12	26	0.014566	6.29E-08
		g	1	2	0.004844	0	1	2	0.003449	0	1	3	0.003646	0
		h	1	2	0.002882	0	1	2	0.005137	0	1	3	0.004788	0
		i	15	16	0.020974	9.74E-08	34	35	0.057051	7.09E-08	21	43	0.020695	6.80E-08
		j	19	20	0.021913	7.22E-08	16	17	0.021173	7.41E-09	14	30	0.022804	7.29E-09
	10,000	a	3	4	0.025311	0	8	10	0.068817	3.73E-09	4	10	0.038389	4.80E-12
		b	2	3	0.014411	0	7	9	0.058538	4.07E-09	4	10	0.035195	3.37E-10
		c	2	3	0.016878	0	7	9	0.05786	4.16E-09	4	10	0.032745	4.09E-12
		d	1	2	0.005907	0	7	9	0.060805	4.32E-09	3	8	0.040586	6.22E-08
		e	1	2	0.007844	0	6	8	0.048647	4.31E-09	4	10	0.037576	0
		f	17	18	0.115479	6.73E-08	12	14	0.100358	3.71E-08	14	30	0.119411	7.57E-09
		g	1	2	0.006684	0	1	2	0.011162	0	1	3	0.008491	0
		h	1	2	0.006599	0	1	2	0.010896	0	1	3	0.008491	0
		i	15	16	0.100351	9.74E-08	35	36	0.315166	9.31E-09	21	43	0.008115	6.63E-08
		j	20	21	0.153714	8.26E-08	18	20	0.147069	5.32E-08	14	30	0.008149	2.07E-08
	100,000	a	3	4	0.156368	0	10	12	0.647913	1.33E-08	4	10	0.008491	1.52E-11
		b	2	3	0.098148	0	8	10	0.487149	1.34E-08	4	10	0.008491	1.07E-09
		c	2	3	0.090468	0	8	10	0.476052	1.38E-08	4	10	0.008499	1.29E-11
		d	1	2	0.038694	0	7	9	0.434821	1.37E-08	4	10	0.008492	0
		e	1	2	0.036204	0	6	8	0.361978	1.36E-08	–	–	–	–
		f	17	18	0.896099	6.73E-08	12	14	0.715902	2.86E-08	14	30	0.008493	2.76E-08
		g	1	2	0.045366	0	1	2	0.067343	0	1	3	0.008491	0
		h	1	2	0.053842	0	1	2	0.081775	0	1	3	0.008439	0
		i	15	16	0.706414	9.74E-08	35	36	2.415865	9.31E-09	21	43	0.008449	6.63E-08
		j	21	22	1.113584	9.08E-08	17	18	0.888265	5.74E-09	14	30	0.008492	7.04E-08
4	1000	a	1	2	0.002586	0	9	11	0.008055	1.55E-08	7	16	0.009909	1.68E-08
		b	2	3	0.003794	0	8	10	0.010698	1.65E-08	5	12	0.008569	5.75E-08
		c	2	3	0.004594	0	8	10	0.007551	1.65E-08	7	16	0.010662	1.19E-08
		d	2	3	0.005619	0	8	10	0.009486	1.65E-08	6	14	0.011261	1.37E-08
		e	2	3	0.003633	0	7	9	0.004957	9.56E-08	–	–	–	–
		f	3	4	0.005493	0	13	15	0.012161	6.18E-08	13	28	0.019133	1.01E-08
		g	2	3	0.007998	0	1	2	0.004058	0	1	3	0.002263	0
		h	2	3	0.006265	0	1	2	0.003805	0	1	3	0.002271	0
		i	14	15	0.013931	4.25E-08	11	12	0.011529	4.69E-08	11	23	0.015554	1.32E-08
		j	19	20	0.019262	5.60E-08	14	16	0.012076	1.47E-08	22	46	0.059155	8.57E-08
	10,000	a	1	2	0.009678	0	9	11	0.049923	4.91E-08	7	16	0.062623	5.32E-08
		b	2	3	0.016263	0	8	10	0.029134	5.22E-08	6	14	0.042619	0
		c	2	3	0.012927	0	8	10	0.030904	5.2E-08	7	16	0.057385	3.78E-08
		d	2	3	0.011865	0	8	10	0.027744	5.22E-08	6	14	0.041896	0
		e	2	3	0.013777	0	8	10	0.027979	3.02E-08	7	16	0.053392	8.98E-09
		f	3	4	0.018759	0	14	16	0.059262	2.11E-08	15	32	0.136765	2.71E-08
		g	2	3	0.01321	0	1	2	0.005562	0	1	3	0.006551	0
		h	2	3	0.016312	0	1	2	0.006919	0	1	3	0.007165	0
		i	14	15	0.078734	4.25E-08	11	12	0.035089	4.69E-08	11	23	0.081224	1.32E-08
		j	20	21	0.111599	6.68E-08	16	17	0.063001	9.6E-08	36	73	0.454409	4.16E-08
	100,000	a	1	2	0.043127	0	12	14	0.310319	1.64E-08	8	18	0.413459	0
		b	2	3	0.081193	0	10	12	0.239006	1.64E-08	6	14	0.311962	0
		c	2	3	0.107093	0	10	12	0.228038	1.65E-08	8	18	0.387133	0
		d	2	3	0.087478	0	9	11	0.200339	1.65E-08	7	16	0.363858	8.87E-09
		e	2	3	0.082591	0	8	10	0.180292	9.56E-08	–	–	–	–
		f	3	4	0.122075	0	13	15	0.375222	1.62E-08	15	32	0.913306	1.07E-08
		g	2	3	0.094085	0	1	2	0.027312	0	1	3	0.026445	0
		h	2	3	0.074505	0	1	2	0.036711	0	1	3	0.023519	0
		i	14	15	0.502872	4.25E-08	11	12	0.346618	4.69E-08	11	23	0.458557	1.32E-08
		j	21	22	0.834635	6.60E-08	21	22	0.539635	2.86E-08	34	69	4.317829	7.32E-09

Table 5.25: Numerical results of AHZP, DLPM and MHZ2 methods for problems 5 & 6

			AHZP				DLPM				MHZ2			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
5	1000	a	21	23	0.027733	7.36E-08	20	22	0.018046	1.9E-08	72	146	0.287652	3.76E-08
		b	21	23	0.029225	8.73E-08	19	21	0.014911	6.5E-08	66	134	0.232369	1.33E-08
		c	20	22	0.030005	3.64E-08	22	24	0.025712	5.2E-08	58	118	0.238636	1.19E-08
		d	20	22	0.028747	6.40E-08	17	19	0.013772	2.86E-08	26	54	0.083404	5.38E-08
		e	19	21	0.029924	8.40E-08	19	21	0.018033	7.86E-08	22	46	0.033511	3.43E-11
		f	19	21	0.027277	6.23E-08	20	22	0.022314	5.24E-08	23	48	0.063364	6.77E-10
		g	20	22	0.025127	4.11E-08	21	23	0.018427	1.33E-08	23	48	0.029344	1.12E-08
		h	20	22	0.030343	3.51E-08	18	20	0.013309	7E-08	24	50	0.041549	1.54E-12
		i	19	21	0.024382	8.14E-08	23	25	0.02337	3.55E-08	24	50	0.032437	2.76E-08
		j	22	24	0.033354	9.37E-08	26	28	0.023005	5.67E-08	97	196	0.36152	7.74E-08
	10,000	a	23	25	0.170924	7.05E-08	19	21	0.117426	6.55E-08	116	234	4.616119	3.02E-14
		b	23	25	0.205809	6.66E-08	21	23	0.129085	9.13E-09	101	204	3.876723	2.81E-08
		c	23	25	0.176271	6.75E-08	23	25	0.113115	1.16E-08	77	156	2.932597	2.24E-08
		d	21	23	0.016398	9.88E-08	21	23	0.123108	8.05E-08	31	64	0.507235	2.16E-08
		e	20	22	0.159046	9.10E-08	18	20	0.100911	2.34E-08	34	70	0.584631	1.79E-08
		f	20	22	0.150213	9.97E-08	21	23	0.132707	9.05E-08	34	70	0.739965	2.47E-08
		g	21	23	0.156448	3.72E-08	21	23	0.119906	2.46E-08	37	76	0.760182	4.08E-08
		h	21	23	0.179718	7.22E-08	19	21	0.103678	3.3E-08	44	90	1.113109	6.57E-08
		i	20	22	0.156408	9.54E-08	23	25	0.124324	7.27E-08	42	86	1.006519	9.35E-08
		j	33	35	0.300055	3.98E-08	25	27	0.168255	6.9E-08	103	208	3.985787	4.52E-11
	100,000	a	29	31	1.871459	8.12E-08	30	32	1.322559	9.57E-09	136	274	44.98661	1.98E-11
		b	33	35	2.222619	8.99E-08	24	26	0.966724	4.65E-08	90	182	24.87766	1.04E-13
		c	22	24	1.259493	8.90E-08	27	29	1.290821	2.81E-09	156	314	49.74681	5.94E-11
		d	27	29	1.757906	5.22E-08	23	25	0.853701	3.56E-09	43	88	6.515603	8.74E-08
		e	21	23	1.529182	8.68E-08	24	26	0.878537	1.41E-08	141	284	43.60521	1.23E-12
		f	22	24	1.282186	3.67E-08	25	27	0.979922	8.2E-08	98	198	24.36554	8.36E-08
		g	23	25	1.533765	7.42E-08	22	24	0.874073	9.47E-08	110	222	30.01365	6.84E-08
		h	27	29	1.618124	5.25E-08	24	26	0.899008	6.51E-08	66	134	16.01713	5.67E-08
		i	22	24	1.599614	3.81E-08	21	23	0.766356	6.1E-08	108	218	31.08076	4.57E-08
		j	48	50	3.614278	4.80E-08	26	28	1.173124	4.63E-08	149	300	49.26632	8.94E-09
6	1000	a	5	6	0.008197	1.99E-08	20	22	0.020082	6.72E-08	5	12	0.015091	1.44E-08
		b	4	6	0.009882	3.38E-08	19	21	0.016673	4.12E-08	4	10	0.012829	7.04E-08
		c	4	5	0.009503	2.85E-08	16	18	0.014398	6.94E-08	3	8	0.012309	5.60E-08
		d	4	6	0.010028	6.94E-08	18	20	0.017263	9.06E-08	4	10	0.011619	1.09E-08
		e	5	6	0.011102	8.70E-09	16	18	0.019424	6.68E-08	5	12	0.013669	1.40E-08
		f	9	11	0.018667	2.66E-08	24	26	0.021211	4.88E-08	14	30	0.037924	1.62E-08
		g	5	6	0.008951	6.53E-09	18	20	0.028041	5.96E-08	5	12	0.011778	1.71E-08
		h	4	6	0.010416	8.38E-08	14	16	0.014701	4.53E-08	4	10	0.018409	4.75E-08
		i	9	11	0.014754	3.89E-08	18	20	0.016558	4.08E-08	10	22	0.020116	8.53E-08
		j	10	12	0.016587	8.20E-08	20	22	0.016348	4.37E-08	13	28	0.038621	1.14E-08
	10,000	a	5	6	0.050829	6.29E-08	21	23	0.132844	8.36E-08	5	12	0.075892	4.55E-08
		b	5	6	0.054451	2.81E-09	20	22	0.101194	5.13E-08	5	12	0.075933	1.54E-08
		c	4	5	0.037826	9.01E-08	17	19	0.084469	8.63E-08	4	10	0.061303	8.85E-10
		d	5	6	0.049487	5.77E-09	20	22	0.101155	4.43E-08	4	10	0.057372	3.46E-08
		e	5	6	0.049131	2.75E-08	17	19	0.088458	8.31E-08	5	12	0.076715	4.43E-08
		f	10	11	0.084716	6.96E-08	20	22	0.101944	6.42E-08	16	34	0.215221	3.14E-08
		g	5	6	0.050138	2.06E-08	19	21	0.094151	7.41E-08	5	12	0.074563	5.42E-08
		h	5	6	0.046593	6.97E-09	15	17	0.078243	5.63E-08	5	12	0.072422	1.04E-08
		i	9	11	0.088093	6.40E-08	18	20	0.090554	7.96E-08	13	28	0.170664	3.47E-08
		j	11	13	0.115064	4.25E-08	21	23	0.107088	5.45E-08	13	28	0.207531	5.05E-08
	100,000	a	5	7	0.368131	4.50E-08	23	25	0.832082	7.69E-08	6	14	0.629553	7.19E-10
		b	5	6	0.306706	8.89E-09	21	23	0.762478	6.38E-08	5	12	0.577905	4.88E-08
		c	4	6	0.305309	6.44E-08	19	21	0.708245	4.22E-08	4	10	0.449126	2.80E-09
		d	5	6	0.321479	1.83E-08	21	23	0.771601	5.51E-08	5	12	0.561066	7.59E-09
		e	5	6	0.342185	8.70E-08	19	21	0.708118	4.07E-08	6	14	0.633854	7.01E-10
		f	10	12	0.772599	4.73E-08	21	23	0.774138	4.62E-08	25	52	2.971307	2.27E-08
		g	5	6	0.329965	6.53E-08	21	23	1.088519	5.12E-08	6	14	0.600771	8.57E-10
		h	5	6	0.315683	2.20E-08	21	23	0.953348	5.42E-08	5	12	0.538802	3.30E-08
		i	8	9	0.668158	4.41E-08	19	21	0.828752	9.17E-08	13	28	1.278113	1.96E-08
		j	12	13	0.941143	7.22E-08	22	24	0.935143	6.81E-08	14	30	1.602602	3.05E-08

Table 5.26: Numerical results of AHZP, DLPM and MHZ2 methods for problem 7

			AHZP				DLPM				MHZ2			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
7	1000	a	1	2	0.005026	0	9	11	0.007589	8E-08	1	3	0.003413	0
		b	1	2	0.006373	0	9	11	0.007307	1.14E-08	1	3	0.004025	0
		c	1	2	0.003898	0	8	10	0.009289	6.58E-08	1	3	0.004507	0
		d	2	3	0.001587	0	8	10	0.005504	3.57E-08	6	14	0.034503	4.32E-08
		e	12	14	0.026804	4.22E-08	7	9	0.008721	1.62E-08	6	14	0.030029	1.16E-08
		f	3	4	0.009235	0	12	14	0.013601	2.29E-08	15	32	0.076977	9.71E-08
		g	1	2	0.002775	0	1	2	0.005216	0	1	3	0.007172	0
		h	2	3	0.005357	0	1	2	0.005377	0	1	3	0.006341	0
		i	13	14	0.038009	7.55E-08	12	13	0.008639	3.43E-08	68	137	4.254011	0
		j	13	15	0.036556	9.78E-08	15	17	0.013004	2.99E-08	14	30	0.067168	2.23E-08
	10,000	a	1	2	0.013503	0	10	12	0.044684	2.53E-08	1	3	0.009628	0
		b	1	2	0.012553	0	9	11	0.041937	3.62E-08	1	3	0.008406	0
		c	1	2	0.008625	0	9	11	0.033604	2.08E-08	1	3	0.007386	0
		d	2	3	0.014651	0	9	11	0.039714	1.13E-08	7	16	0.199211	5.07E-09
		e	13	14	0.156798	7.26E-08	7	9	0.026131	5.12E-08	6	14	0.167163	3.66E-08
		f	3	4	0.047113	0	14	15	0.050996	2.79E-08	19	40	0.514222	5.87E-09
		g	1	2	0.011784	0	1	2	0.006773	0	1	3	0.026011	0
		h	2	3	0.016919	0	1	2	0.006426	0	1	3	0.025715	0
		i	13	14	0.155533	7.55E-08	12	13	0.042429	3.43E-08	52	105	33.66117	0
		j	14	16	0.170662	6.40E-08	16	18	0.068509	2.53E-08	–	–	–	–
	100,000	a	1	2	0.087812	0	12	14	0.328656	3.55E-08	1	3	0.064636	0
		b	1	2	0.084601	0	10	12	0.289284	4.09E-08	1	3	0.049102	0
		c	1	2	0.051268	0	10	12	0.264591	2.54E-08	1	3	0.039347	0
		d	2	3	0.173898	0	10	12	0.267096	1.53E-08	7	16	1.393001	1.60E-08
		e	13	15	1.314268	9.11E-08	8	10	0.281714	1.62E-08	7	16	1.353115	4.30E-09
		f	3	4	0.272453	0	14	15	0.382704	2.68E-08	15	32	3.503018	3.80E-09
		g	1	2	0.078522	0	1	2	0.041848	0	1	3	0.191312	0
		h	2	3	0.106994	0	1	2	0.048667	0	1	3	0.180528	0
		i	13	14	1.066867	7.55E-08	12	13	0.305448	3.43E-08	52	105	278.5842	0
		j	15	17	1.597813	4.39E-08	21	23	0.679073	1.87E-08	21	43	4.950861	2.38E-08

Table 5.27: Summary of test results reported in Table 5.23-5.26

Methods	Iter	Percentage	Funeval	Percentage	Time (s)	Percentage
AHZP	109	51.91%	138	65.71%	115	54.76%
DLPM	29	13.81%	48	22.86%	66	31.43%
MHZ2	15	7.14%	3	1.43%	29	13.81%
Undecided	57	27.14%	21	10%	0	0%

Even though, AHZP,DLPM, and MHZ2 methods have successfully solved the seven test problems reported in the numerical experiment, but the AHZP method converges faster to the solution of (1.1) than the other methods because it has the least number of iterations. Moreover, the MHZ2 method failed during the iteration process by clear indication from Table 5.24. From Table 5.24, MHZ2 failed in 100,000 dimension with initial point $x5$ for problem 3. In addition, from 5.24, MHZ2 failed in 1000 and 100,000 dimensions with the same initial guess $x5$ for problem 4. For the function evaluations presented in Tables 5.23-5.26, when compared to the DLPM and MHZ2 methods, it is obvious that the AHZP method has the fewest number of function evaluations, as expected. Furthermore, the numerical results presented in Tables 5.23-5.26 clearly demonstrate that the AHZP method solves all test problems in less CPU time than the DLPM and MHZ2 methods. In fact, the proposed method outperforms the DLPM and MHZ2 methods for nearly all problems because it has the fewest number of iterations and processor time, as well as the fewest number of function evaluations.

Tables 5.27 summarized the results from Tables 5.23-5.26 to show which approach is best in terms of the number of iterations, function evaluations, and Processor time. From Table 5.27, the AHZP method wins with least number of iteration in about 51.91% of the problems (that is, 109 out of 210). However, DLPM and MHZ2 methods solve 13.81% (29 out of 210) and 7.14% (15 out of 210) of the problems respectively. Meanwhile, the results summary shows that at least two of the three methods have the same number of iterations on 57 out of 210 problems, i.e., 27.14% of the problems and reported as undecided in Table 5.27. For the function evaluations, the AHZP method solves 65.71% (138 out of 210) of the problems, whereas the DLPM and MHZ2 methods solve 22.86% (48 out of 210) and 1.43% (3 out of 210) of the problems, respectively. However, 10% (21 out of 210) reported being undecided. In terms of CPU time, DLPM and MHZ2 methods solve 31.43% (66 out of 210) and 13.81% (29 out of 210) of the problems respectively. The AHZP method on the other hand solve 54.76% (115 out of 210) of the problems.

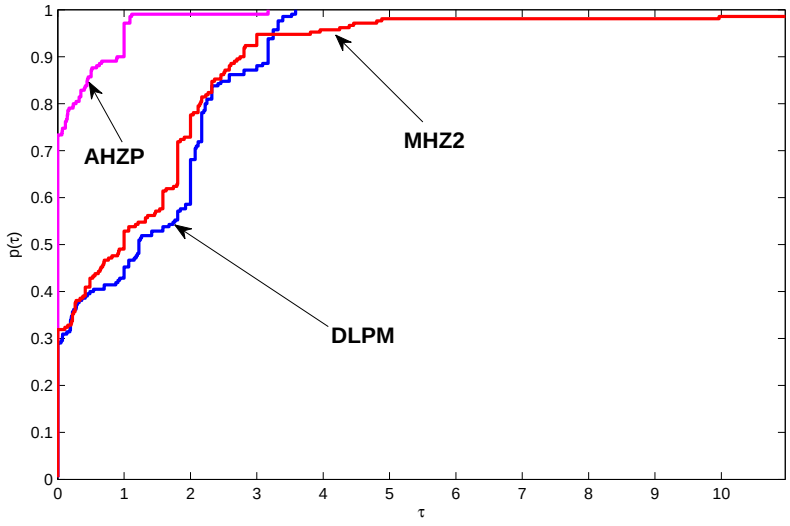

Figure 5.12: Performance profile for the number of iterations

Dolan and Moré [44] evaluation tool was used to present a graphical view of each of the three methods used in the experiments to interpret the results presented in Tables 5.23-5.26. Figures 5.12-5.14 depict the performance profiles of the AHZP, DLPM, and MHZ2 methods in terms of iteration numbers, number of function evaluations, and CPU time. For each of the three figures, a fraction p(τ) of the problems considered is plotted for which a method is within a factor τ of the best time. Each of the three figures has a top curve that specifically corresponds to the AHZP scheme. We conclude that our proposed algorithm is more effective than the compared ones for solving large-scale nonlinear monotone equations with convex constraint, based on the results from Tables 5.23-5.26 and Figures 5.12-5.14.

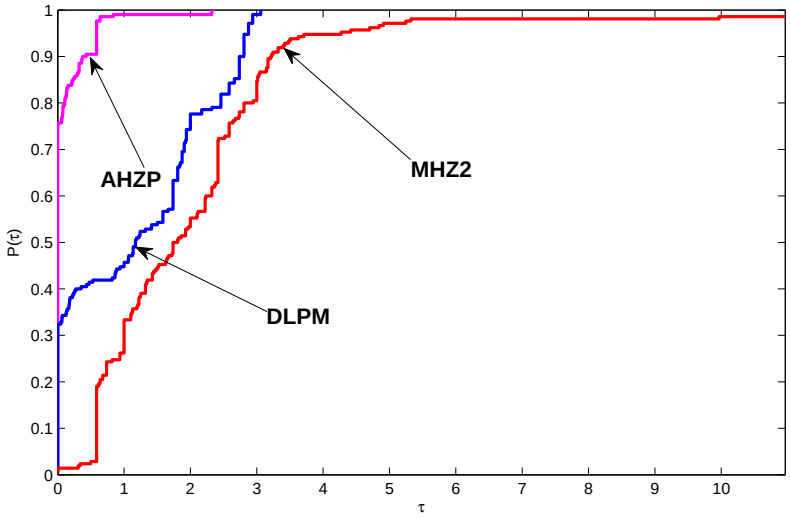

Figure 5.13: Performance profile for the functions evaluations

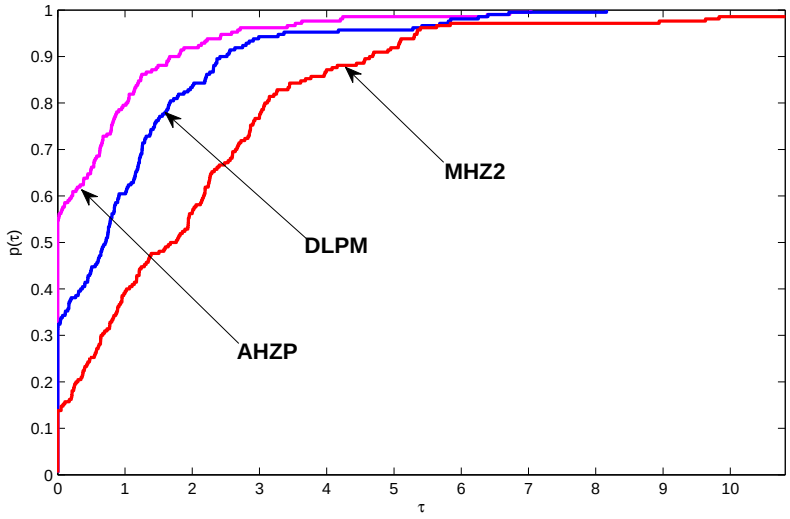

Figure 5.14: Performance profile for the CPU time (in second)

5.5 Applications of AHZP Algorithm in compressive sensing

In this subsection, the AHZP algorithm is successfully applied to handle the ℓ_1-norm regularization problem (2.39) in image recovery. This is made possible by solving the monotone nonlinear system of equations (2.45). Numerical tests were performed to demonstrate the effectiveness of the AHZP method in restoring certain blurred images by comparing it with SGCS method [119]. When implementing AHZP algorithm in this experiments, and the following parameters are set $\xi = 1$, $\sigma = 10^{-4}$, $\rho = 0.5$, $\tau = 0.4$, and $\zeta = 1.3$. The parameters of SGCS algorithm, are taken as in [119]. The iteration is set to stop for both methods if the following conditions occur:

$$\frac{|f(x_k) - f(x_{k-1})|}{|f(x_{k-1})|} < 10^{-5},$$

with a merit function $f(x)$ define as $f(x) = \frac{1}{2}\|\vartheta - Mx\|_2^2 + \pi\|x\|_1$,. In addition, during the image de-blurring experiment, the codes were ran with $x_0 = M^T\vartheta$, as initial point. Signal-to-Noise Ratio (SNR) is defined by

$$\text{SNR} = 20 \times \log_{10}\left(\frac{\|\hat{x}\|}{\|\tilde{x} - \hat{x}\|}\right),$$

where, $\hat{x}$ and $\tilde{x}$ are the original image and the restored image respectively. Furthermore, Structural Similarity (SSIM) index is used in this paper in order to measure the quality of the restored images [106]. The MATLAB implementation of the SSIM index can be obtained at `http://www.cns.nyu.edu/~lcv/ssim/`.

Table 5.28: Numerical results for AHZP and SGCS in image restoration

Images	Size	AHZP				SGCS			
		Iter	Time (s)	SNR	SSIM	Iter	Time (s)	SNR	SSIM
Figure 5.15 (Halilu)	256×256	31	4.44	30.31	0.92	200	20.78	30.43	0.92
Figure 5.16 (Tajmahal)	256×256	30	4.48	23.26	0.85	399	42.58	23.41	0.87
Figure 5.17 (LPU Mall)	256×256	40	5.91	20.03	0.81	701	72.86	20.02	0.82
Figure 5.18 (Duck)	256×256	32	4.88	20.84	0.89	345	35.53	20.77	0.90

Figure 5.15: The original image (first row first column), the blurred image (first row second column), the restored image by AHZP (second row first column) and SGCS (second row second column

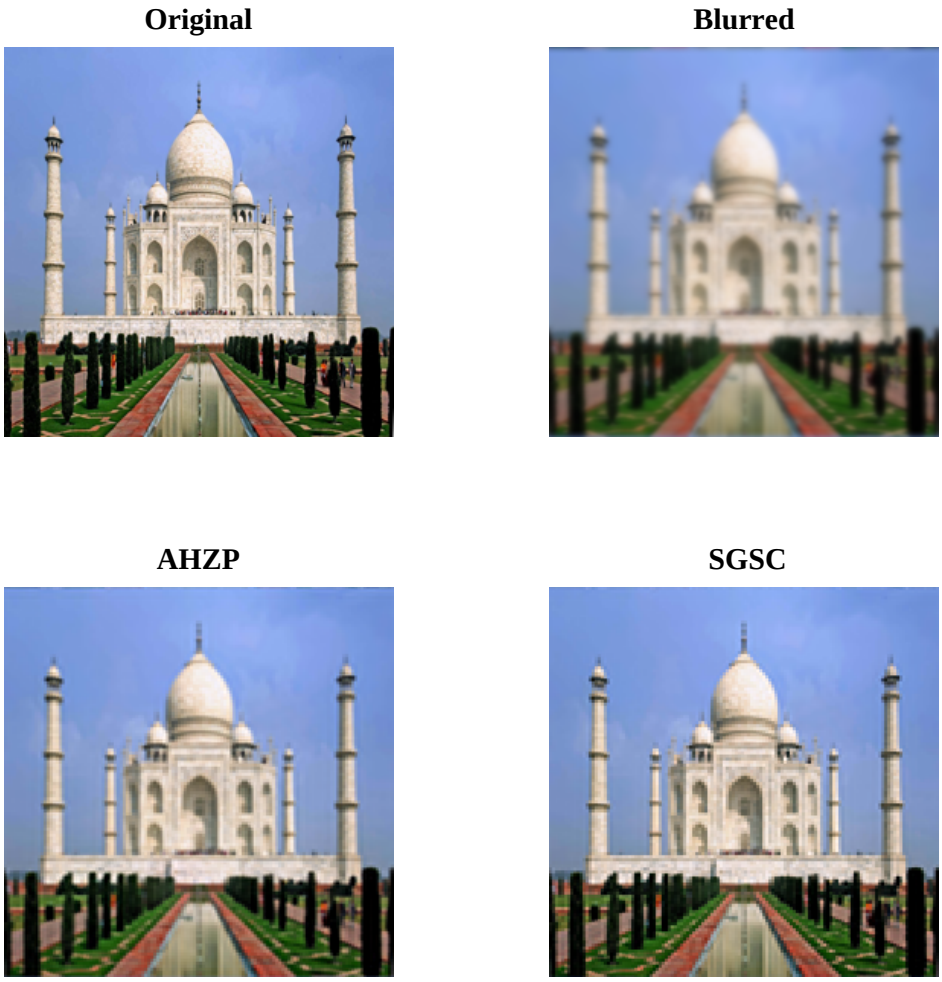

Figure 5.16: The original image (first row first column), the blurred image (first row second column), the restored image by AHZP (second row first column) and SGCS (second row second column

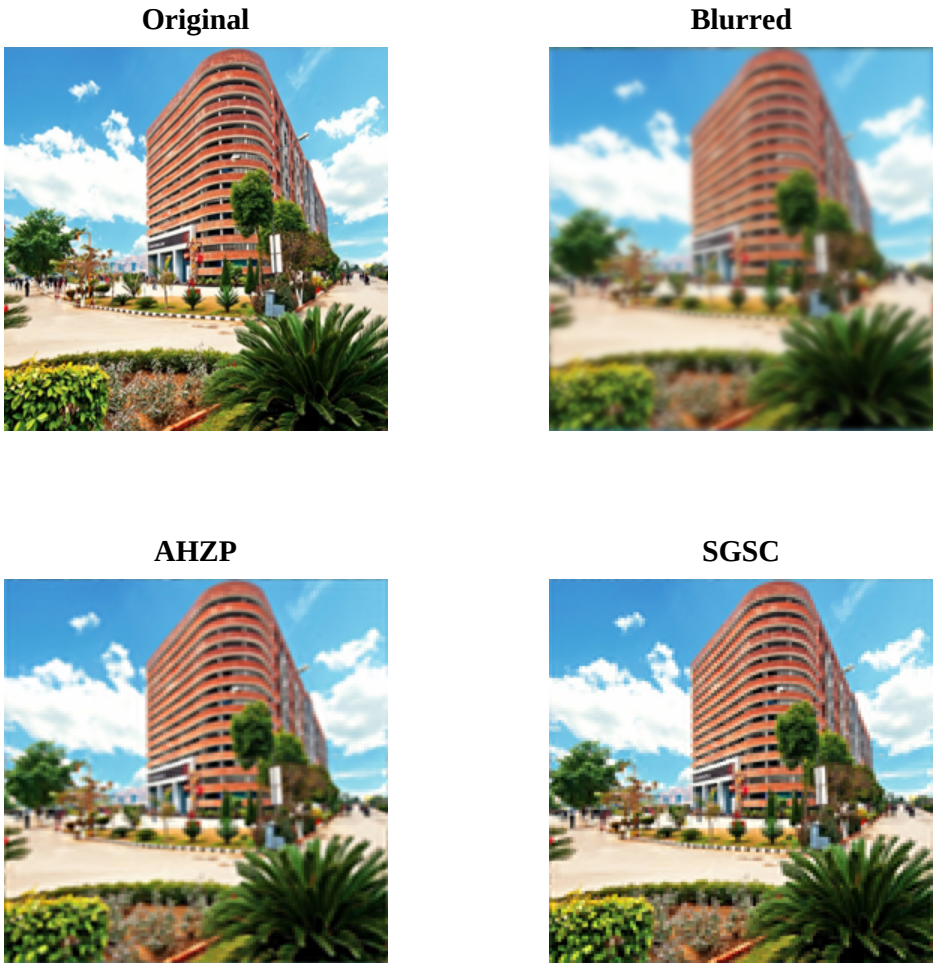

Figure 5.17: The original image (first row first column), the blurred image (first row second column), the restored image by AHZP (second row first column) and SGCS (second row second column

Figure 5.18: The original image (first row first column), the blurred image (first row second column), the restored image by AHZP (second row first column) and SGCS (second row second column

Figures 5.15-5.18 are generated to show the restoration results of different images obtained by AHZP and SGCS methods. Both methods are successful in restoring all four images, but the data in Table 5.28 clearly show that the proposed method has a higher efficacy. Despite the fact that the SGCS method has higher SNR values than the AHZP method in the images in Figures 5.15 and 5.16, the AHZP method has higher SNR values than the SGCS method in the images in Figures 5.17 and 5.18. Figure 5.15 shows that both methods have the same SSIM values. Even though, the SSIM values of the SGCS method in Figures 5.16-5.18 are higher than those of the AHZP method. However, the AHZP method has the least number of iteration and CPU time than the SGCS method. These results demonstrate that the proposed method is more effective at restoring blurred images than SGCS method.

5.5.1 Conclusion

Accelerated Hager-Zhang Projection Method with application in image deblurring problems is presented in this chapter. This is achieved by proposing a new choice of the Hager-Zhang non negative parameter. Numerical comparisons were performed using large-scale test problems, and the AHZP method outperformed the DLPM [1] and MHZ2 [93] methods as shown in Tables 5.23-5.26 and Figures 5.12-5.14. Furthermore, the AHZP method is successfully applied to deal with the experiments on the ℓ_1-norm regularization problem in image restoration and compared its performance with the SGCS method [119]. The experiments were carried out on various samples of images (Figures 5.15-5.18) and the results are recorded in Table 5.28, which clearly show that the AHZP approach is more effective.

CHAPTER SIX

On Solving Double Direction Methods For Convex Constrained Monotone Nonlinear Equations with Image Restoration

6.1 Introduction

Vast applications of derivative-free methods to restore the blurred images in compressive sensing has become an important trend in recent years. This research, a double direction method for better image restoration is proposed. Besides this, two double direction algorithms to solve constrained monotone nonlinear equations (1.3) are presented. Besides this, two double direction algorithms to solve constrained monotone nonlinear equations are presented. The main idea employed in the first algorithm is to approximate the Jacobian matrix via acceleration parameter in order to propose an effective derivative-free method. The second algorithm involve hybridizing the scheme of the first algorithm with Picard-Mann hybrid iterative scheme (2.18). In addition, the step length is calculated using inexact line search technique. The proposed methods are proven to be globally convergent under some mild conditions . The numerical experiment, shown in this paper, depicts the efficiency of the proposed methods. Furthermore, the second method is successfully applied to handle the ℓ_1-norm regularization problem in image recovery which exhibits a better result than the existing method in the previous literature. Throughout this chapter, the space $\mathbb{R}^n$ denote the $n-$dimensional real space equipped with the Euclidean norm $\|\cdot\|$, $F_k = F(x_k)$, and $F'_k = F'(x_k)$.

6.2 Derivation of the proposed methods

Now, to present the iterative scheme of our methods, we suggest the search directions d_k^i and d_k^{ii} in (2.16) to be respectively defined as:

$$d_k^i = -\gamma_k^{-1} F_k, \tag{6.179}$$

and

$$d_k^{ii} = -F_k, \tag{6.180}$$

where $\gamma_k > 0$. By putting (6.179) and (6.180) in to (2.16) we obtained

$$x_{k+1} = x_k - (\alpha_k + \alpha_k^2 \gamma_k)\gamma_k^{-1} F_k. \tag{6.181}$$

The main idea of our scheme is to derive the update of the acceleration parameter γ_k. This is made possible by using the first order Taylor's expansion given by

$$F_{k+1} \approx F_k + F'(\xi)(x_{k+1} - x_k). \tag{6.182}$$

By multiplying (6.182) through by τ_k we have

$$\tau_k F_{k+1} \approx \tau_k F_k + \tau_k F'(\xi)(\alpha_k + \alpha_k^2 \gamma_k) d_k^i, \tag{6.183}$$

where τ_k is defined in (2.15) and the parameter ξ fulfills the conditions $\xi \in [x_k, x_{k+1}]$,

$$\xi = x_k + \delta(x_{k+1} - x_k), \quad 0 \leq \delta \leq 1. \tag{6.184}$$

Taking $\delta = 1$ in (6.184) and obtained $\xi = x_{k+1}$.

Therefore we are interested to approximate the Jacobian matrix via

$$\tau_k F'(\xi) \approx \gamma_{k+1} I. \tag{6.185}$$

Now, from (6.183) and (6.185) its not difficult to verify the modified secant equation:

$$\gamma_{k+1} s_k = \tau_k y_k, \tag{6.186}$$

where, $y_k = F_{k+1} - F_k$, $s_k = (\alpha_k + \alpha_k^2\gamma_k)d_k^i$.

By multiplying y_k^T to the both side of (6.186) the proposed acceleration parameter is defined as:

$$\gamma_{k+1} = \frac{\|s_k\|^2\|y_k\|^2}{(\alpha_k + \alpha_k^2\gamma_k)^2(y_k^T d_k)^2}. \tag{6.187}$$

From (6.179) and (6.181) we have our first iterative scheme as:

$$x_{k+1} = x_k + (\alpha_k + \alpha_k^2\gamma_k)d_k^i. \tag{6.188}$$

Algorithm 5: Derivative-free double direction method (DDDM)

Input: Given $x_0 \in \Omega$, $d_0^i = -F_0$, $\rho \in (0,1)$, $\sigma > 0$, $\varepsilon = 10^{-6}$, $\gamma_0 = 1$, and $\xi > 0$, set $k = 0$.

Step 1: Compute F_k. If $\|F_k)\| \leq \varepsilon$ then stop, else go to Step 2.

Step 2: Let $\mu_k = (\alpha_k + \alpha_k^2\gamma_k)$ where, $\alpha_k = \xi\rho^{m_k}$, with m_k being the smallest nonnegative integer m such that

$$-F(x_k + \mu_k d_k^i)^T d_k^i \geq \sigma\mu_k\|d_k^i\|^2. \tag{6.189}$$

Step 3: Set $z_k = x_k + \mu_k d_k^i$.

Step 4: If $z_k \in \Omega$ and $\|F(z_k)\| \leq \varepsilon$, stop, otherwise go to Step 5.

Step 5: Compute the next iterate by

$$x_{k+1} = P_\Omega[x_k - \lambda_k F(z_k)], \tag{6.190}$$

where, $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step 6: Determine $\gamma_{k+1} = \dfrac{\|s_k\|^2\|y_k\|^2}{(\alpha_k + \alpha_k^2\gamma_k)^2(y_k^T d_k^i)^2}$, where $s_k = z_k - x_k$ and $y_k = F(z_k) - F(x_k)$.

Step 7: Compute the search direction d_{k+1}^i using (6.179).

Step 8: Consider $k = k+1$ and go to Step 2.

To present the hybrid type of derivative-free double direction method, the mapping T in (2.18) is redefined by $T(v_k) = v_k - (\alpha_k + \alpha_k^2\gamma_k)\gamma_k^{-1}F_k$. By this definition and (2.18) we have

$$\begin{cases} x_1 = x \in \Omega, \\ v_k = (1-\eta_k)x_k + \eta_k T(x_k) = x_k - \eta_k(\alpha_k + \alpha_k^2\gamma_k)\gamma_k^{-1}F_k, \\ x_{k+1} = T(v_k) = v_k - (\alpha_k + \alpha_k^2\gamma_k)\gamma_k^{-1}F_k, \quad k \in \mathbb{N}. \end{cases} \tag{6.191}$$

From the second and third equations in (6.191) we obtain the second iterative scheme,

$$x_{k+1} = x_k - t_k(\alpha_k + \alpha_k^2\gamma_k)\gamma_k^{-1}F_k, \tag{6.192}$$

where, $t_k = (\eta_k + 1)$ and $\eta_k \in (0,1)$ is a correction parameter. We however choose $\eta = \eta_k$, so that $t = (\eta + 1) \in (1,2)$ From (6.192), we can easily show that, the search direction is defined as:

$$d_k = \begin{cases} -F_k, & \text{if} \quad k = 0, \\ -t\gamma_k^{-1}F_k, & \text{if} \quad k \geq 1. \end{cases} \tag{6.193}$$

Algorithm 6: Derivative-free double direction method (HDDM)

Input: Given $x_0 \in \Omega$, $\gamma_0 = 1$, $d_0 = -F_0$, $t \in (1,2)$, $\rho \in (0,1)$, $\sigma > 0$, $\varepsilon = 10^{-6}$, and $\xi > 0$, set $k = 0$.

Step 1: Compute F_k. If $\|F_k\| \leq \varepsilon$ then stop, else go to Step 2.

Step 2: Let $\mu_k = (\alpha_k + \alpha_k^2 \gamma_k)$ where, $\alpha_k = \xi \rho^{m_k}$, with m_k being the smallest nonnegative integer m such that

$$-F(x_k + \mu_k d_k)^T d_k \geq \sigma \mu_k \|d_k\|^2. \tag{6.194}$$

Step 3: Set $z_k = x_k + \mu_k d_k$.

Step 4: If $z_k \in \Omega$ and $\|F(z_k)\| \leq \varepsilon$, stop, otherwise go to Step 5.

Step 5: Compute the next iterate by

$$x_{k+1} = P_\Omega[x_k - \lambda_k F(z_k)], \tag{6.195}$$

where, $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step 6: $\gamma_{k+1} = \dfrac{\|s_k\|^2 \|y_k\|^2}{(\alpha_k + \alpha_k^2 \gamma_k)^2 (y_k^T d_k)^2}$.

Step 7: Compute the search direction $d_{k+1} = -t \gamma_{k+1}^{-1} F_{k+1}$.

Step 8: Consider $k = k+1$ and go to Step 2.

Remark 6.2.1 *We give the following remarks*

(a) From (3.50), we have

$$y_{k-1}^T s_{k-1} = s_{k-1}^T (F(z_{k-1}) - F(x_{k-1})) \geq c \|s_{k-1}\|^2 > 0. \tag{6.196}$$

From (3.49) and Cauchy-Schwarz inequality, we have

$$y_{k-1}^T s_{k-1} = (F(z_{k-1}) - F(x_{k-1}))^T (z_{k-1} - x_{k-1}) \leq L \|s_{k-1}\|^2. \tag{6.197}$$

Also, from (6.196) and (6.197) it is simple to check that, $L\|s_{k-1}\|^2 \geq y_{k-1}^T s_{k-1} \geq c\|s_{k-1}\|^2$, *this implies that* $\frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \leq \frac{1}{c}$ *and* $\frac{\|s_{k-1}\|^2}{y_{k-1}^T s_{k-1}} \geq \frac{1}{L}$. *This means* τ_k *in*

(2.15) *is well defined. Therefore, we have*

$$\frac{1}{L} \leq \tau_k \leq \frac{1}{c}. \tag{6.198}$$

(b) From assumption (A1) we have

$$\|y_{k-1}\| = \|F(z_{k-1}) - F(x_{k-1})\| \leq L\|s_{k-1}\| = L\mu_{k-1}\|d_{k-1}\|. \tag{6.199}$$

(c) From assumption (A2) we have

$$y_{k-1}^T d_{k-1} = (F(z_{k-1}) - F(x_{k-1}))^T \frac{s_{k-1}}{\mu_{k-1}} \geq \frac{c\|s_{k-1}\|^2}{\mu_{k-1}} = c\mu_{k-1}\|d_{k-1}\|^2 > 0. \tag{6.200}$$

Since $y_{k-1}^T d_{k-1} > 0$, the proposed γ_k in (6.187) is well defined.

(d) From assumption (A1) and (A2) we have

$$\gamma_k = \frac{\|y_{k-1}\|^2\|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2} \leq \frac{L^2\|s_{k-1}\|^4}{c^2\|s_{k-1}\|^4} = \frac{L^2}{c^2}. \tag{6.201}$$

6.3 Convergence Analysis of Algorithm 6 (HDDM)

The convergence analysis of HDDM Algorithm is established in this subsection.

Lemma 6.3.1 *Suppose assumptions (A1)-(A3) hold and $\{x_k\}$ be generated by Algorithm 6, then the search direction d_k fulfills the descent property i.e.,*

$$F_k^T d_k < 0. \tag{6.202}$$

Proof For $k = 0$, from (6.193) we have $F_0^T d_0 = -\|F_0\|^2 < 0$. For $k \geq 1$, since $t \in (1,2)$, from (6.187) and (6.193), we have,

$$F_k^T d_k = -\frac{t(y_k^T s_{k-1})^2}{\|y_{k-1}\|^2\|s_{k-1}\|^2}\|F_k\|^2 < 0. \tag{6.203}$$

■

Lemma 6.3.2 *Suppose assumptions (A1)-(A3) hold and let $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm 6, then $\{x_k\}$ and $\{z_k\}$ are bounded. In addition, we have*

$$\|d_k\| \leq M. \tag{6.204}$$

$$\lim_{k\to\infty} \|z_k - x_k\| = 0. \tag{6.205}$$

$$\lim_{k\to\infty} \|x_k - x_{k+1}\| = 0. \tag{6.206}$$

Proof The proofs of (6.205) and (6.206) follow from Lemma 3.3.3. As a result, there are positive constants m_1 and κ such that

$$\|F(x_k)\| \leq m_1, \tag{6.207}$$

$$\|F(z_k)\| \leq \kappa, \tag{6.208}$$

and

$$\lim_{k\to\infty} \alpha_k \|d_k\| = 0. \tag{6.209}$$

Now,for $k = 0$, by (6.193) and (6.207) we have $\|d_0\| = -\|F_0\| < m_1$. For $k \geq 1$, from (6.193) (6.207) and remarks (a)-(d) we have

$$\|d_k\| = \left\| \frac{-t(y_{k-1}^T s_{k-1})^2 F_k}{\|y_{k-1}\|^2 \|s_{k-1}\|^2} \right\| \leq \frac{t\|y_{k-1}\|^2 \|s_{k-1}\|^2 \|F_k\|}{\|y_{k-1}\|^2 \|s_{k-1}\|^2} \leq t\|F_k\| \leq tm_1.$$

Taking $M = tm_1$, we have (6.204). ∎

Theorem 6.3.3 *Suppose assumption (A1)-(A3) hold and $\{x_k\}$ be generated by Algorithm 6. Then*

$$\liminf_{k\to\infty} \|F_k\| = 0. \tag{6.210}$$

Proof For the sake of contradiction, suppose that (6.210) is not true, then there exists $n_0 > 0$ such that

$$\|F_k\| \geq \delta_0 \quad \text{holds,} \quad \forall k > 0. \tag{6.211}$$

Suppose that the search direction $\|d_k\| \neq 0$ unless at the solution, then there is a constant, say δ_1

$$\|d_k\| \geq \delta_1. \tag{6.212}$$

If $\mu_k \neq \xi$, then by the definition of $\mu_k = (\alpha_k + \alpha_k^2\gamma_k)$, $\rho^{-1}\mu_k$ does not satisfy (6.194) i.e.,

$$-F(x_k + \rho^{-1}\mu_k d_k)^T d_k < \sigma\rho^{-1}\mu_k\|d_k\|^2.$$

Now, combining with (6.203) gives

$$\begin{aligned}
\frac{t(y_{k-1}^T s_{k-1})^2}{\|y_{k-1}\|^2\|s_{k-1}\|^2}\|F(x_k)\|^2 &= -F_k^T d_k, \\
&= (F(x_k + \rho^{-1}\mu_k d_k) - F(x_k))^T d_k - F(x_k + \rho^{-1}\mu_k d_k)^T d_k, \\
&\leq L\rho^{-1}\mu_k\|d_k\|^2 + \sigma\rho^{-1}\mu_k\|d_k\|^2. \\
&= \mu_k\|d_k\|(L+\sigma)\rho^{-1}\|d_k\|. \qquad (6.213)
\end{aligned}$$

Therefore, from (6.213), we have

$$\begin{aligned}
\mu_k\|d_k\| &\geq \frac{t(y_{k-1}^T s_{k-1})^2}{\|y_{k-1}\|^2\|s_{k-1}\|^2}\frac{\rho\|F_k\|^2}{(L+\sigma)\|d_k\|} \\
&\geq \frac{tc^2\|s_{k-1}\|^4}{\|y_{k-1}\|^2\|s_{k-1}\|^2}\frac{\rho\|F_k\|^2}{(L+\sigma)\|d_k\|} \\
&\geq \frac{tc^2\|s_{k-1}\|^4}{L^2\|s_{k-1}\|^4}\frac{\rho\|F_k\|^2}{(L+\sigma)\|d_k\|} \\
&\geq \frac{tc^2}{L^2}\frac{\rho\delta_0^2}{(L+\sigma)M}. \qquad (6.214)
\end{aligned}$$

The second and third inequalities follow from (6.196) and (6.200), respectively. The last inequality follows from (6.204) and (9.302).

The inequality (6.214) contradicts (6.209). Therefore (6.210) holds. Hence, the proof is complete. ■

Remark 6.3.4 *If the correction parameter $\eta = 0$, then the convergence result of DDDM Algorithm follows.*

6.4 Numerical Experiments

Some numerical results are provided in the first part of this subsection, to show the effectiveness of our methods i.e.,

- Algorithm 5: Derivative-free double direction method (DDDM).
- Algorithm 6: Hybrid derivative-free double direction method (HDDM).

Since the proposed algorithms are derivative-free, therefore their performances are compared with the following derivative-free method existing in the literature.

- A descent derivative-free algorithm for nonlinear monotone equations with convex constraints (DDPM) [79].

In the second part, HDDM algorithm is applied to solve ℓ_1-norm regularization problem in image deblurring problems. The computer codes used are written in Matlab 8.3.0 (R2014a) and run on a personal computer equipped with a 1.80 GHz CPU processor and 8 GB RAM. When implementing the algorithms in this experiments, the following parameters are set in our algorithms; $\xi = 1$, $\sigma = 10^{-4}$ and $\rho = 0.9$, we however set $t = 1.2$, in HDDM algorithm. The parameters of DDPM algorithm, are taken as in [79]. The iteration is set to stop for all the three methods if the following conditions occur: (i) when $\|F(x_k)\| \leq 10^{-6}$, (ii) when $\|F(z_k)\| \leq 10^{-6}$, or (iii) when the iterations exceed 1000 but no point of x_k satisfying the stopping criterion is obtained. We have tried the three methods on the previous six test problems with different initial guess and dimension (n values). To show the extensive numerical experiments of DDDM, HDDM and DDPM methods, the experimentation was carried out with three different dimensions namely, 1000, 50,000 and 100,000. The reported numerical results of the three (3) methods are shown in Tables 6.30–6.35. From the Tables, "IT" represents the total number of iterations, "TIME(s)" represents the CPU time (in seconds), "IP" represents the initial points and "FVAL" represents the functions evaluations, and "NORM" represents $\|F_k\|$ at the termination point.

The following initial points were used for the test problems:

Table 6.29: Starting points used to test problems

INITIAL POINTS (IP)	VALUES
$x1$	$(1,1,...,1)^T$
$x2$	$\left(\frac{1}{2},\frac{1}{2^2},...,\frac{1}{2^n}\right)^T$
$x3$	$\left(0,\frac{1}{2},\frac{2}{3},...,1-\frac{1}{n}\right)^T$
$x4$	$\left(1,\frac{1}{2},\frac{1}{3},...,\frac{1}{n}\right)^T$
$x5$	$(2,2,...,2)^T$
$x6$	$\left(\frac{1}{4},\frac{-1}{4},...,\frac{(-1)^n}{4}\right)^T$

The following test problems were used in the experiments:

Problem 1 [12]

$F_1(x) = e^{x_1} - 1,$

$F_i(x) = e^{x_i} + x_{i-1} - 1, \quad i = 2,3,\ldots,n-1,$

where $\Omega = \mathbb{R}^n_+$.

Problem 2 [79]

$F_i(x) = log(x_i + 1) - \frac{x_i}{n}, \quad i = 2,3,\ldots,n-1,$

where $\Omega = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i > -1, \quad i = 1,2,\ldots,n\}$.

Problem 3 [12]

$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Clearly, Problem 3 is nonsmooth at $x = 0$.

Problem 4 [79]

$F_i(x) = e^{x_i} - 1, \quad i = 1,2,\ldots,n,$

where $\Omega = \mathbb{R}^n_+$.

Problem 5 [12]

$F_i(x) = 2x_i - \sin|x_i - 1|, \quad i = 1,2,\ldots,n,$

where $\Omega = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i > -1, \quad i = 1,2,\ldots,n\}$.

Clearly, Problem 5 is nonsmooth at $x = 1$.

Problem 6 [12]

$F_1 = x_1 - e^{\cos\left(\frac{x_1+x_2}{n+1}\right)},$

$F_i = x_i - e^{\cos\left(\frac{x_{i-1}+x_i+x_{i+1}}{n+1}\right)},$

$F_n = x_n - e^{\cos\left(\frac{x_{n-1}+x_n}{n+1}\right)}, \quad i = 2,3,\ldots,n-1,$

where $\Omega = \mathbb{R}^n_+$.

Table 6.30: Numerical results of DDDM, HDDM and DDPM methods for problem 1

		DDDM				HDDM				DDPM			
DIMENSION	IP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	6	8	0.01579	1.14E-07	4	6	0.021898	7.32E-07	16	18	0.013754	3.57E-07
	x2	9	10	0.017381	7.65E-08	5	7	0.012606	4.74E-07	17	18	0.008548	4.51E-07
	x3	7	9	0.015226	1.69E-07	5	7	0.015344	5.46E-07	16	18	0.011436	3.58E-07
	x4	8	10	0.015319	8.82E-08	7	9	0.016014	1.97E-08	16	18	0.009857	5.85E-07
	x5	6	8	0.012441	1.41E-07	4	6	0.0137	7.15E-07	16	18	0.01057	7.28E-07
	x6	5	7	0.011342	2E-07	4	6	0.009925	6.42E-08	1	2	0.003141	0
50,000	x1	6	8	0.234366	3.56E-07	4	6	0.194617	5.71E-08	19	21	0.242217	8.67E-07
	x2	9	10	0.283763	7.65E-08	5	7	0.181244	4.74E-07	17	18	0.19226	4.51E-07
	x3	6	8	0.241063	3.8E-07	4	6	0.210873	3.96E-07	19	21	0.247919	8.66E-07
	x4	8	10	0.265288	8.81E-08	7	9	0.24463	1.97E-08	16	18	0.199716	5.85E-07
	x5	7	9	0.249878	6.29E-08	5	7	0.204272	5.25E-08	22	24	0.285129	6.24E-07
	x6	5	7	0.13476	1.98E-07	4	6	0.130291	6.11E-08	1	2	0.027491	0
100,000	x1	6	8	0.48908	4.55E-07	4	6	0.340777	7.89E-08	21	23	0.478644	4.22E-07
	x2	9	10	0.499119	7.65E-08	5	7	0.344758	4.74E-07	17	18	0.343464	4.51E-07
	x3	6	8	0.425237	4.74E-07	4	6	0.321529	2.81E-07	21	23	0.480251	4.22E-07
	x4	8	10	0.537506	8.81E-08	7	9	0.5049	1.97E-08	16	18	0.380719	5.85E-07
	x5	7	9	0.599913	8.9E-08	5	7	0.386616	7.43E-08	24	26	0.598374	5.53E-07
	x6	5	7	0.248611	1.98E-07	4	6	0.219518	6.11E-08	1	2	0.051124	0

Table 6.31: Numerical results of DDDM, HDDM and DDPM methods for problem 2

		DDDM				HDDM				DDPM			
DIMENSION	IP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	6	8	0.035927	4.76E-07	5	7	0.008075	1.4E-07	24	26	0.020136	5.08E-07
	x2	5	7	0.013451	9.65E-07	5	7	0.010812	9.2E-08	18	20	0.014231	6.32E-07
	x3	13	15	0.021244	5.11E-07	16	18	0.029107	2.45E-08	24	26	0.021171	5.03E-07
	x4	7	9	0.01367	2.3E-07	7	9	0.011141	4.5E-07	20	22	0.016412	5.2E-07
	x5	6	8	0.010638	4.59E-07	5	7	0.011358	1.35E-07	25	27	0.019453	7.65E-07
	x6	5	7	0.012076	3.97E-07	4	6	0.011281	1.19E-07	22	24	0.02353	7.68E-07
50,000	x1	7	9	0.243708	1.86E-07	5	7	0.181388	8.25E-07	28	30	0.494609	5.79E-07
	x2	5	7	0.156054	9.31E-07	5	7	0.170949	9.13E-08	18	20	0.319672	6.29E-07
	x3	18	20	0.551114	2.99E-07	88	90	2.626965	1.53E-07	28	30	0.541877	5.79E-07
	x4	7	9	0.212973	2.25E-07	7	9	0.23127	7.61E-07	20	22	0.352528	5.18E-07
	x5	7	9	0.215001	1.75E-07	5	7	0.18158	7.64E-07	29	31	0.528937	7.28E-07
	x6	6	8	0.211737	1.57E-07	4	6	0.16431	7.01E-07	25	27	0.458729	6.71E-07
100,000	x1	7	9	0.442629	2.63E-07	6	8	0.504699	2.72E-08	29	31	0.97942	6.35E-07
	x2	5	7	0.314231	9.31E-07	5	7	0.369069	9.13E-08	18	20	0.588916	6.29E-07
	x3	18	20	1.001162	2.16E-07	75	77	4.725202	3.04E-07	29	31	0.99589	6.34E-07
	x4	7	9	0.39876	2.25E-07	7	9	0.434612	7.66E-07	20	22	0.658945	5.18E-07
	x5	7	9	0.419536	2.46E-07	6	8	0.410346	2.52E-08	30	32	1.029389	5.14E-07
	x6	6	8	0.3679	2.21E-07	4	6	0.304972	9.89E-07	25	27	0.86033	9.49E-07

Table 6.32: Numerical results of DDDM, HDDM and DDPM methods for problem 3

		DDDM				HDDM				DDPM			
DIMENSION	IP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	6	8	0.015569	9.07E-08	4	6	0.009941	5.25E-07	24	26	0.014296	7.56E-07
	x2	5	7	0.007412	1.51E-07	4	6	0.007669	4.34E-08	19	21	0.011837	5.31E-07
	x3	6	8	0.011093	1.11E-07	4	6	0.008335	6.63E-07	24	26	0.014005	7.54E-07
	x4	4	6	0.00622	5.3E-07	4	6	0.00931	7.72E-07	20	22	0.009989	5.75E-07
	x5	6	8	0.010519	1.51E-07	4	6	0.008023	8.52E-07	23	25	0.015776	7.62E-07
	x6	1	2	0.00584	0	1	2	0.004187	0	1	2	0.003839	0
50,000	x1	6	8	0.14953	6.42E-07	5	7	0.144335	8.67E-08	28	30	0.310846	7.61E-07
	x2	5	7	0.147519	1.51E-07	4	6	0.112941	4.34E-08	19	21	0.215895	5.31E-07
	x3	6	8	0.154206	6.45E-07	5	7	0.147688	8.73E-08	28	30	0.34001	7.61E-07
	x4	4	6	0.122533	5.25E-07	4	6	0.119342	7.72E-07	20	22	0.225217	5.76E-07
	x5	7	9	0.187996	6.51E-08	5	7	0.146349	1.41E-07	31	33	0.357249	6.27E-07
	x6	1	2	0.049631	0	1	2	0.049287	0	1	2	0.032177	0
100,000	x1	6	8	0.28952	9.07E-07	5	7	0.257853	1.23E-07	29	31	0.628223	7.94E-07
	x2	5	7	0.220009	1.51E-07	4	6	0.241617	4.34E-08	19	21	0.406173	5.31E-07
	x3	6	8	0.275188	9.1E-07	5	7	0.255319	1.23E-07	29	31	0.606163	7.94E-07
	x4	4	6	0.192486	5.25E-07	4	6	0.224371	7.72E-07	20	22	0.412978	5.76E-07
	x5	7	9	0.29537	9.21E-08	5	7	0.259356	1.99E-07	33	35	0.789034	7.38E-07
	x6	1	2	0.087854	0	1	2	0.098928	0	1	2	0.057458	0

Table 6.33: Numerical results of DDDM, HDDM and DDPM methods for problem 4

		DDDM				HDDM				DDPM			
DIMENSION	IP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	5	7	0.013752	7.83E-07	5	7	0.008646	6.87E-08	25	27	0.012583	5.99E-07
	x2	5	7	0.007757	2.22E-07	4	6	0.008513	9.53E-08	18	20	0.008533	9.22E-07
	x3	7	9	0.006986	4.87E-07	5	7	0.008774	2.43E-07	25	27	0.01383	5.99E-07
	x4	6	8	0.011449	1.13E-07	7	9	0.012199	1.12E-07	22	24	0.016128	6.5E-07
	x5	6	8	0.012186	4.38E-07	5	7	0.008843	8.4E-08	26	28	0.013009	5.98E-07
	x6	1	2	0.004079	0	1	2	0.004508	0	1	2	0.002897	0
50,000	x1	6	8	0.128542	3.37E-07	5	7	0.123549	4.85E-07	29	31	0.292743	5.1E-07
	x2	5	7	0.106402	2.22E-07	4	6	0.097253	9.53E-08	18	20	0.180574	9.22E-07
	x3	6	8	0.134465	4.31E-07	5	7	0.127108	5.23E-07	29	31	0.318688	5.1E-07
	x4	6	8	0.1664	1.13E-07	7	9	0.192778	1.11E-07	22	24	0.225365	6.5E-07
	x5	7	9	0.152226	1.88E-07	5	7	0.136408	5.94E-07	31	33	0.2956	9.63E-07
	x6	1	2	0.02421	0	1	2	0.0219	0	1	2	0.015788	0
100,000	x1	6	8	0.25947	4.76E-07	5	7	0.243199	6.87E-07	30	32	0.543538	5.07E-07
	x2	5	7	0.191936	2.22E-07	4	6	0.177432	9.53E-08	18	20	0.309274	9.22E-07
	x3	6	8	0.223481	5.48E-07	5	7	0.256258	7.15E-07	30	32	0.568598	5.07E-07
	x4	6	8	0.223235	1.13E-07	7	9	0.293253	1.11E-07	22	24	0.372303	6.5E-07
	x5	7	9	0.281456	2.66E-07	5	7	0.240241	8.4E-07	33	35	0.659843	8.31E-07
	x6	1	2	0.038523	0	1	2	0.038031	0	1	2	0.028365	0

Table 6.34: Numerical results of DDDM, HDDM and DDPM methods for problem 5

		DDDM				HDDM				DDPM			
DIMENSION	IP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	7	9	0.016248	8.05E-07	6	8	0.012379	3.59E-07	16	18	0.01181	7.39E-07
	x2	7	9	0.017203	8.13E-07	6	8	0.012694	4.1E-07	16	18	0.012994	8.26E-07
	x3	8	10	0.018875	1.9E-07	7	9	0.014774	2.79E-07	16	18	0.011449	7.31E-07
	x4	9	11	0.017352	2.53E-07	8	10	0.016785	9.87E-08	16	18	0.012042	8.17E-07
	x5	7	9	0.012254	9.93E-07	6	8	0.011468	4.4E-07	16	18	0.013446	6.79E-07
	x6	6	8	0.013715	1.26E-07	5	7	0.011815	9.4E-08	14	16	0.00959	5.51E-07
50,000	x1	8	10	0.279344	6.81E-07	7	9	0.287708	1.87E-07	18	20	0.276878	6.33E-07
	x2	8	10	0.282651	6.69E-07	7	9	0.273813	1.94E-07	18	20	0.330285	7.09E-07
	x3	8	10	0.279367	7.52E-07	8	10	0.314475	9.55E-08	18	20	0.297498	6.33E-07
	x4	10	12	0.379323	6.93E-07	10	12	0.360856	1.02E-07	18	20	0.294785	7.09E-07
	x5	8	10	0.271348	8.39E-07	7	9	0.264383	2.29E-07	20	22	0.316913	3.5E-07
	x6	6	8	0.223637	8.92E-07	5	7	0.248618	6.64E-07	19	21	0.296125	3.88E-07
100,000	x1	8	10	0.599236	9.63E-07	7	9	0.503587	2.65E-07	19	21	0.590877	7.35E-07
	x2	8	10	0.556636	9.45E-07	7	9	0.549131	2.74E-07	19	21	0.555972	3.49E-07
	x3	9	11	0.631565	1.22E-07	8	10	0.620913	1.52E-07	19	21	0.556349	7.35E-07
	x4	10	12	0.683173	9.98E-07	13	15	0.905601	1.32E-07	19	21	0.595335	3.49E-07
	x5	9	11	0.590531	1.42E-07	7	9	0.482389	3.24E-07	20	22	0.596198	4.95E-07
	x6	7	9	0.478583	1.51E-07	5	7	0.428161	9.4E-07	19	21	0.567323	9.66E-07

Table 6.35: Numerical results of DDDM, HDDM and DDPM methods for problem 6

		DDDM				HDDM				DDPM			
DIMENSION	IP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	6	8	0.018569	1.66E-07	4	6	0.00993	9.72E-07	25	27	0.019235	8.09E-07
	x2	6	8	0.013581	2.63E-07	5	7	0.012451	3.58E-08	26	28	0.019958	6.40E-07
	x3	6	8	0.014415	1.67E-07	4	6	0.011838	9.77E-07	25	27	0.020869	8.13E-07
	x4	6	8	0.010919	2.63E-07	5	7	0.012611	3.57E-08	26	28	0.024025	6.38E-07
	x5	6	8	0.015888	6.95E-08	4	6	0.010797	4.06E-07	24	26	0.018638	6.76E-07
	x6	6	8	0.013901	2.87E-07	5	7	0.011791	3.91E-08	26	28	0.023805	6.99E-07
50,000	x1	7	9	0.262561	7.20E-08	5	7	0.225942	1.62E-07	30	32	0.545067	8.07E-07
	x2	7	9	0.267122	1.14E-07	5	7	0.252171	2.57E-07	33	35	0.618142	5.83E-07
	x3	7	9	0.280711	7.21E-08	5	7	0.222714	1.62E-07	30	32	0.545653	8.08E-07
	x4	7	9	0.300448	1.14E-07	5	7	0.238128	2.57E-07	33	35	0.632667	5.83E-07
	x5	6	8	0.24229	4.95E-07	5	7	0.22748	6.79E-08	27	29	0.475108	5.98E-07
	x6	7	9	0.270401	1.24E-07	5	7	0.243771	2.81E-07	33	35	0.619143	9.87E-07
100,000	x1	7	9	0.580075	1.02E-07	5	7	0.412347	2.30E-07	32	34	1.222911	8.23E-07
	x2	7	9	0.53134	1.61E-07	5	7	0.423014	3.63E-07	36	38	1.466238	5.96E-07
	x3	7	9	0.49702	1.02E-07	5	7	0.458689	2.30E-07	32	34	1.252447	8.23E-07
	x4	7	9	0.545991	1.61E-07	5	7	0.41576	3.63E-07	36	38	1.469043	5.96E-07
	x5	6	8	0.489506	7.00E-07	5	7	0.45221	9.60E-08	28	30	1.018146	7.96E-07
	x6	7	9	0.496382	1.76E-07	5	7	0.456999	3.97E-07	37	39	1.537679	6.04E-07

From Tables 6.30–6.35 one can easily see that, the three methods are trying to solve the problem in (1.3), but the improved efficacy of the proposed methods are clear from the tables. In fact, the HDDM approach significantly outperforms the DDDM method for nearly all the problems assessed, since it has the least number of iterations, functions evaluations and CPU time, which are below the number of iterations, functions evaluations and the CPU time for the DDDM methods. This is apparently due to the contribution of correction parameter η in the HDDM iteration. Furthermore, the numerical results reported in Tables 6.30–6.35 were evidently showed that HDDM and DDDM methods, are far better than DDPM method, because they have number of iterations, functions evaluations and CPU time, that far below the ones for DDPM method.

We generate Figures 6.19–6.21 using the performance profiles of Dolan and Moré [44] to show the performance of each of the three methods. This is done by plotting the fraction $p(\tau)$ of the problems for which each method is within τ of the smallest number of iterations, CPU time and function evaluations respectively. Observe that, from Figures 6.19–6.21, the curves corresponding to the HDDM and DDDM methods remain above the other curves representing the DDPM method. This shows that the methods proposed in this paper, outperforms the method compared in terms of fewer iterations, functions evaluation and CPU time (in seconds).

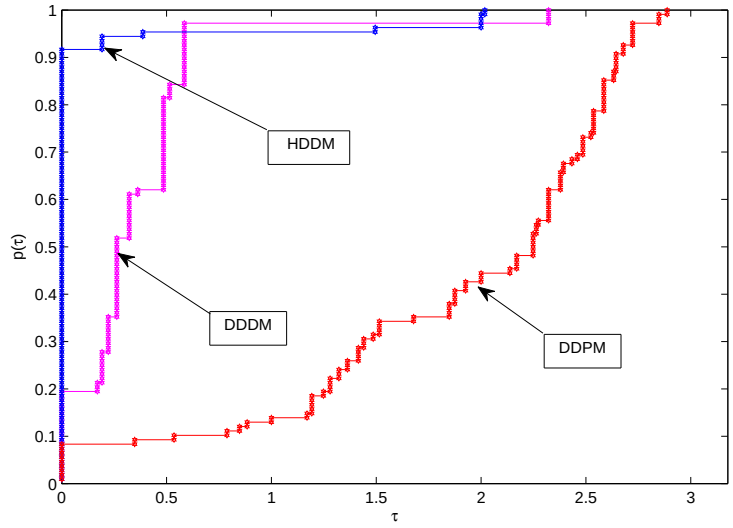

Figure 6.19: Performance profile for the number of iterations

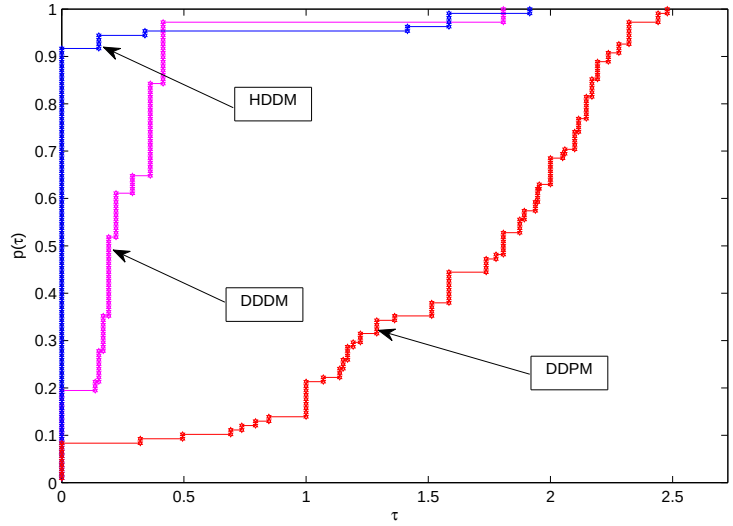

Figure 6.20: Performance profile for the functions evaluations

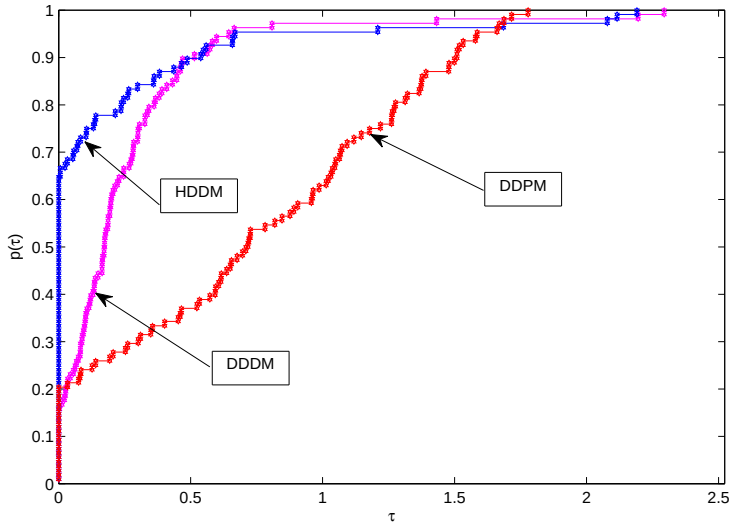

Figure 6.21: Performance profile for the CPU time (in second)

However, the curves in the figures indicated that HDDM methods is the most efficient among the three method. Finally, from three Figures and the results in Tables 6.30–6.35 it is obvious that our approaches are very successful in solving large-scale nonlinear problems.

6.5 Applications of Algorithm 6 in Image Deblurring

This subsection is dedicated to handle the ℓ_1-norm regularization problem (2.39) in image deblurring problems. To further highlight the performance of the HDDM scheme, some numerical experiments was carried out to restore some blurred images by comparing it with a spectral gradient method for ℓ_1-norm problems in compressed sensing (SGCS) [119]. When implementing HDDM algorithm in this experiments, and the following parameters are set $\xi = 1$, $\sigma = 10^{-4}$, $\rho = 0.9$, and $t = 1.2$. The parameters of SGCS algorithm, are taken as in [119]. The iteration

is set to stop for both methods if the following conditions occur:

$$\frac{|f(x_k) - f(x_{k-1})|}{|f(x_{k-1})|} < 10^{-5},$$

with a merit function $f(x)$ define as $f(x) = \frac{1}{2}\|\vartheta - Mx\|_2^2 + \pi\|x\|_1$,. In addition, during the image de-blurring experiment, the codes were ran with $x_0 = M^T\vartheta$, as initial point. Signal-to-Noise Ratio (SNR) defined by

$$\text{SNR} = 20 \times \log_{10}\left(\frac{\|\hat{x}\|}{\|x - \hat{x}\|}\right),$$

where, $\hat{x}$ and x are the original image and the restored image respectively. Furthermore, Structural Similarity (SSIM) index is used in this paper in order to measure the quality of the restored images [106]. The MATLAB implementation of the SSIM index can be obtained at `http://www.cns.nyu.edu/~lcv/ssim/`.

Table 6.36: Numerical results for HDDM and SGCS in image restoration

		HDDM				SGCS			
IMAGE	SIZE	ITER	TIME(s)	SNR	SSIM	ITER	TIME (s)	SNR	SSIM
Lena	256×256	26	1.03	24.41	0.90	270	10.84	24.14	0.90
House	256×256	23	0.91	27.66	0.89	228	9.17	27.50	0.89
Barbara	512×512	28	6.11	19.64	0.79	257	16.22	19.68	0.79
Pepper	256×256	26	0.97	22.84	0.87	296	12.02	22.80	0.87

Figure 6.22: Performance profile for the CPU time (in second)

Figure 6.22 is generated to show the restoration results of different images obtained by HDDM and SGCS methods. From Table 6.36, both methods are successful in restoration of all the four images, but the improved efficacy of the proposed method is very clear from the table. Although, both method have the same values of SSIM, but SNR values reported in the table indicated that HDDM method is slightly restored the four blurred images with better quality than SGCS method except for Barbara. In fact, the HDDM method remarkably outperforms the SGCS method for restoring all the images assessed, since it has the least number of iterations and CPU time (in second), which are far below the number of iterations and the CPU time for the SGCS method. These results show that the proposed method can restore corrupted images efficiently.

6.6 Conclusion

In this chapter, two double direction methods (DDDM and HDDM) for convex constrained monotone nonlinear equations are presented. The DDDM scheme was achieved by proposing the acceleration parameter that approximated the Jacobian with diagonal matrix in a sufficient manner. We further improverd the numerical performances of DDDM by applying Picard-Mann hybrid iterative process [95], and derived HDDM scheme. The proposed methods are completely matrix-free that are globally convergent under certain appropriate conditions. Numerical comparisons have been made using large-scale test problems. Furthermore, Tables 6.30–6.35 and Figures 6.19–6.21 showed that the proposed methods are practically quite welcome, because they have the least number of iteration and CPU time compared to DDPM method [79]. In addition, HDDM method is successfully applied to deal with the experiments on the ℓ_1-norm regularization problem in image restoration and compared its performance with SGCS method [119]. The experiments were carried out with various samples of images, as shown in Table 6.36, which clearly demonstrated a higher efficiency for the HDDM method. Finally, Figure 4, indicated that the HDDM method had better restoration of the blurred images because it had the lowest mean iteration number and CPU time than SGCS method.

CHAPTER SEVEN

Three term Hager-Zhang Projection Method for Monotone Nonlinear Equations

7.1 Introduction

The importance of a nonlinear system of equations in real life cannot be emphasized, as it emerges regularly in mathematical problems in domains such as science, engineering, and industry. One of the applications of nonlinear equations is the Chandrasekhar integral equation, which is useful in radiactive transfer [109]. It also has applications in the problem arising from motion control involving two–joint planar robotic manipulator. [**?**].

In this chapter, a conjugate gradient method with the projection technique of Solodov and Svaiter [Kluwer Academic Publishers, (1998), pp. 355-369] to solve monotone nonlinear equations is presented. The proposed method improved the numerical performance of the Hager-Zhang method proposed by Waziri et al. [Applied Mathematics and Computation, 361 (2019), pp. 645–660], by extending its direction to three-term conjugate gradient direction. The proposed method has been shown to be globally convergent under some mild conditions. Numerical experiments show that the proposed method is effective and produces better results than existing methods. Throughout this chapter, the space $\mathbb{R}^n$ denote the $n-$dimensional real space equipped with the Euclidean norm $\|\cdot\|$, $F_k = F(x_k)$, and $F'_k = F'(x_k)$.

7.2 Three-term Hager-Zhang method

Details of the proposed method has been presented in this section. To begin, let us define the hyperplane projection technique by Solodov and Svaiter in [97]. Let the sequence $\{z_k\}$ is generated via $z_k = x_k + \alpha_k d_k$, $\alpha_k > 0$ such that

$$(x_k - z_k)^T F(z_k) > 0. \tag{7.215}$$

Since the mapping F is monotone, then for each of solution x^* of (1.1), we have

$$(x^* - z_k)^T F(z_k) = (x^* - z_k)^T (F(z_k) - F(x^*)) \leq 0. \tag{7.216}$$

From (7.215) and (7.216) is easy to see that the hyperplane

$$H_k = \{x \in \mathbb{R}^n | (x - z_k)^T F(z_k) = 0\}, \tag{7.217}$$

separates the current iterate x_k strictly from the solution x^* of (1.1). The authors in [97] suggested that the next iterate x_{k+1} should be the projection of x_k onto the hyperplane H_k and determined

$$x_{k+1} = x_k - \frac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2} F(z_k). \tag{7.218}$$

Now, consider the Hager-Zhang method proposed by Waziri et al.[112], with CG direction given by

$$d_k = \begin{cases} -F_k, & \text{if} \quad k = 0, \\ -F_k + \beta_k^{HZ} d_{k-1}, & \text{if} \quad k \geq 1, \end{cases} \tag{7.219}$$

where the Hager-Zhang CG parameter is defined as

$$\beta_k^{HZ}(\theta) = \frac{F_k^T y_{k-1}}{s_{k-1}^T y_{k-1}} - \theta_k \frac{\|y_{k-1}\|^2 F_k^T s_{k-1}}{(y_{k-1}^T s_{k-1})^2}, \tag{7.220}$$

the Hager-Zhang nonnegative parameter θ_k is given as

$$\theta_k = p - q\frac{(y_{k-1}^T s_{k-1})^2}{\|s_{k-1}\|^2\|y_{k-1}\|^2}, \quad p \geq \frac{1}{4}, \quad \text{and} \quad q \leq 0.$$

From (7.219) and (7.220) it was shown that the search direction is given by

$$d_k = -Q_k F_k, \quad k \geq 1. \tag{7.221}$$

where Q_k is the search direction matrix given by

$$Q_k = I - \frac{s_{k-1}y_{k-1}^T}{y_{k-1}^T s_{k-1}} + \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}s_{k-1}^T}{(y_{k-1}^T s_{k-1})^2}. \tag{7.222}$$

It was also shown that d_k in [112] meets the property $d_k^T F_k \leq -\lambda_k^- \|F_k\|^2$, where $\lambda_k^- > 0$. This is vital property that makes this method to be globally convergent.

Motivated by the above idea and the approach of three-term CG method presented by Awwal et al. [13]. We are interested in defining the search direction as

$$d_k = -A_k F_k, \quad k \geq 1, \tag{7.223}$$

where A_k is the symmetric search direction matrix given by

$$A_k = \frac{Q_k + Q_k^T}{2} = I - \frac{1}{2}\frac{s_{k-1}y_{k-1}^T + y_{k-1}s_{k-1}^T}{y_{k-1}^T s_{k-1}} + \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}s_{k-1}^T}{(y_{k-1}^T s_{k-1})^2}. \tag{7.224}$$

Substituting (7.224) into (7.223) gives

$$d_k = -F_k + \left(\frac{1}{2}\frac{y_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} - \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}^T F_k}{(y_{k-1}^T s_{k-1})^2}\right) s_{k-1} + \frac{1}{2}\frac{s_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} y_{k-1}. \tag{7.225}$$

The proposed search direction d_k in (7.225) can be expressed as a three-term CG direction, as shown below.

$$d_k = \begin{cases} -F_k, & \text{if} \quad k = 0, \\ -F_k + \beta_k^{TTHZP} s_{k-1} + \gamma_k y_{k-1}, & \text{if} \quad k \geq 1, \end{cases} \tag{7.226}$$

where $\beta_k^{TTHZP} = \left(\frac{1}{2} \frac{y_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} - \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}^T F_k}{(y_{k-1}^T s_{k-1})^2} \right)$, $\gamma_k = \frac{1}{2} \frac{s_{k-1}^T F_k}{y_{k-1}^T s_{k-1}}$, and $\theta_k = p - q \frac{(y_{k-1}^T s_{k-1})^2}{\|s_{k-1}\|^2 \|y_{k-1}\|^2}$, $p \geq \frac{1}{4}$, and $q \leq 0$. This clearly indicates that $\theta_k \geq \frac{1}{4}$ $\forall k$.

Lemma 7.2.1 *The search direction d_k defined by (7.226) satisfies the property*

$$F_k^T d_k \leq -b\|F_k\|^2, \quad b > 0. \tag{7.227}$$

for all $k \geq 0$.

Proof For $k = 0$, from (7.226) we have $F_0^T d_0 = -\|F_0\|^2 < 0$. For $k \geq 1$, we have,

$$\begin{aligned} F_k^T d_k &= -\|F_k\|^2 + \frac{1}{2} \frac{(y_{k-1}^T F_k)(s_{k-1}^T F_k)}{y_{k-1}^T s_{k-1}} + \frac{1}{2} \frac{(y_{k-1}^T F_k)(s_{k-1}^T F_k)}{y_{k-1}^T s_{k-1}} - \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}^T F_k}{(y_{k-1}^T s_{k-1})^2}, \\ &= -\|F_k\|^2 + \frac{(y_{k-1}^T F_k)(s_{k-1}^T F_k)}{y_{k-1}^T s_{k-1}} - \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}^T F_k}{(y_{k-1}^T s_{k-1})^2}, \\ &= \frac{-\|F_k\|^2 (y_{k-1}^T s_{k-1})^2 + (y_{k-1}^T s_{k-1})(y_{k-1}^T F_k)(s_{k-1}^T F_k) - \theta_k \|y_{k-1}\|^2 s_{k-1}^T F_k}{(y_{k-1}^T s_{k-1})^2}. \end{aligned}$$

By applying the inequality $a_1^T a_2 \leq \frac{1}{2}(\|a_1\|^2 + \|a_2\|^2)$ to the second term of the last equality with $a_1 = \frac{(y_{k-1}^T s_{k-1}) F_k}{\sqrt{2\theta_k}}$ and $a_2 = \sqrt{2\theta_k}(s_{k-1}^T F_k) y_{k-1}$, we obtain $F_k^T d_k \leq -\left(1 - \frac{1}{4\theta_k}\right) \|F_k\|^2$.

Taking $b = \left(1 - \frac{1}{4\theta_k}\right)$, we have (7.227). ■

The following is a description of the proposed method's algorithm

Algorithm 7: Accelerated Hager-Zhang projection Method (TTHZP)

Input: Given $x_0 \in \mathbb{R}^n$, $d_0 = -F_0$, $\rho \in (0,1)$, $\sigma > 0$, $\varepsilon > 0$, and $\xi > 0$, set $k = 0$.

Step 1: Compute F_k.

Step 2: If $\|F_k\| \leq \varepsilon$ then stop, else go to Step 3.

Step 3: Let $\alpha_k = \xi \rho^{m_k}$, with m_k being the smallest nonnegative integer m such that

$$-F(z_k)^T d_k \geq \sigma \alpha_k \|d_k\|^2. \tag{7.228}$$

Step 4: Set $z_k = x_k + \alpha_k d_k$.

Step 5: If $\|F(z_k)\| \leq \varepsilon$, stop, otherwise go to Step 6.

Step 6: Set $x_{k+1} = x_k - \lambda_k F(z_k)$,
where, $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step 7: Compute the search direction d_{k+1} using (7.226).

Step 8: Consider $k = k+1$ and go to Step 2.

Remark 7.2.2 *We give the following remarks*

(a) From (3.50), we have

$$y_{k-1}^T s_{k-1} = s_{k-1}^T (F(z_{k-1}) - F(x_{k-1})) \geq c\|s_{k-1}\|^2 > 0. \tag{7.229}$$

(b) From (3.49) we have

$$\|y_{k-1}\| = \|F(z_{k-1}) - F(x_{k-1})\| \leq L\|s_{k-1}\|. \tag{7.230}$$

7.3 Convergence Analysis of Algorithm 7 (TTHZP)

This subsection provides an analysis of the global convergence of the TTHZP Algorithm.

Lemma 7.3.1 *Suppose assumptions (A1)-(A3) hold and let $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm 7, then $\{x_k\}$ and $\{z_k\}$ are bounded. In addition, we have*

$$\|d_k\| \leq M. \tag{7.231}$$

$$\lim_{k\to\infty} \|z_k - x_k\| = 0. \tag{7.232}$$

$$\lim_{k\to\infty} \|x_k - x_{k+1}\| = 0. \tag{7.233}$$

Proof To show the boundedness of the sequences $\{x_k\}$ and $\{z_k\}$, let $\bar{x} \in S^x$ be any solution of (1.1). Then by monotonicity of F we can write

$$(x_k - \bar{x})^T F(z_k) \geq (x_k - z_k)^T F(z_k). \tag{7.234}$$

Using the line search condition (7.228) and definition of z_k, we have

$$(x_k - z_k)^T F(z_k) \geq \sigma \alpha_k^2 \|d_k\|^2 > 0. \tag{7.235}$$

Also, using $x_{k+1} = x_k - \lambda_k F(z_k)$, we have

$$\begin{aligned}
\|x_{k+1} - \bar{x}\|^2 &= \|x_k - \lambda_k F(z_k) - \bar{x}\|^2 \\
&= \|x_k - \bar{x}\|^2 - 2\lambda_k (x_k - \bar{x})^T F(z_k) + \lambda_k^2 \|F(z_k)\|^2 \\
&\leq \|x_k - \bar{x}\|^2 - 2\lambda_k (x_k - z_k)^T F(z_k) + \lambda_k^2 \|F(z_k)\|^2 \\
&= \|x_k - \bar{x}\|^2 - \frac{((x_k - z_k)^T F(z_k))^2}{\|F(z_k)\|^2} \\
&\leq \|x_k - \bar{x}\|^2,
\end{aligned} \tag{7.236}$$

for which we obtain

$$\|x_{k+1} - \bar{x}\| \leq \|x_k - \bar{x}\|. \tag{7.237}$$

This recursively implies that $\|x_k - \bar{x}\| \leq \|x_0 - \bar{x}\|$, for all k. Clearly, $\{\|x_k - \bar{x}\|\}$ is a decreasing sequence, which implies that $\{x_k\}$ is bounded. Furthermore, utilizing assumption (A1), (A3) and (7.237) we have

$$\|F_k\| = \|F_k - F(\bar{x})\| \leq L\|x_k - \bar{x}\| \leq L\|x_0 - \bar{x}\|. \tag{7.238}$$

Let $m_1 = L\|x_0 - \bar{x}\|$, then we have

$$\|F_k\| \leq m_1. \tag{7.239}$$

From (7.226) and Cauchy–Schwarz inequality, we have

$$|\theta_k| = \left| p - q\frac{(y_{k-1}^T s_{k-1})^2}{\|y_{k-1}\|^2\|s_{k-1}\|^2} \right| \leq p + |q|\frac{\|y_{k-1}\|^2\|s_{k-1}\|^2}{\|y_{k-1}\|^2\|s_{k-1}\|^2} = p + |q| = Q. \tag{7.240}$$

From (7.226),(7.229), (7.230), and (7.240) we have

$$\begin{aligned}
\|d_k\| &= \| - F_k + \beta_k^{TTHZ} s_{k-1} + \gamma_k y_{k-1}\| \\
&= \left\| F_k + \left(\frac{1}{2}\frac{y_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} - \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}^T F_k}{(y_{k-1}^T s_{k-1})^2} \right) + \frac{1}{2}\frac{s_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} \right\| \\
&\leq \|F_k\| + \frac{1}{2}\left\| \frac{y_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} s_{k-1} \right\| + \left\| \theta_k \frac{\|y_{k-1}\|^2 s_{k-1}^T F_k}{(y_{k-1}^T s_{k-1})^2} s_{k-1} \right\| + \frac{1}{2}\left\| \frac{s_{k-1}^T F_k}{y_{k-1}^T s_{k-1}} y_{k-1} \right\| \\
&\leq \|F_k\| + \frac{1}{2}\frac{|y_{k-1}^T F_k|}{y_{k-1}^T s_{k-1}}\|s_{k-1}\| + |\theta_k|\frac{\|y_{k-1}\|^2 |s_{k-1}^T F_k|}{(y_{k-1}^T s_{k-1})^2}\|s_{k-1}\| + \frac{1}{2}\frac{|s_{k-1}^T F_k|}{y_{k-1}^T s_{k-1}}\|y_{k-1}\| \\
&\leq \|F_k\| + \frac{1}{2}\frac{\|y_{k-1}\|\|s_{k-1}\|\|F_k\|}{y_{k-1}^T s_{k-1}} + |\theta_k|\frac{\|y_{k-1}\|^2\|s_{k-1}\|^2\|F_k\|}{(y_{k-1}^T s_{k-1})^2} + \frac{1}{2}\frac{\|y_{k-1}\|s_{k-1}\|\|F_k\|}{y_{k-1}^T s_{k-1}} \\
&\leq \|F_k\| + \frac{\|y_{k-1}\|\|s_{k-1}\|\|F_k\|}{y_{k-1}^T s_{k-1}} + |\theta_k|\frac{\|y_{k-1}\|^2\|s_{k-1}\|^2\|F_k\|}{(y_{k-1}^T s_{k-1})^2} \\
&\leq \|F_k\| + \frac{L\|\|s_{k-1}\|^2\|F_k\|}{c\|s_{k-1}\|^2} + Q\frac{L^2\|\|s_{k-1}\|^4\|F_k\|}{c^2\|s_{k-1}\|^4} \\
&\leq \|F_k\| + \frac{L}{c}\|F_k\| + Q\frac{L^2}{c^2}\|F_k\| \\
&= \left(1 + \frac{L}{c} + Q\frac{L^2}{c^2}\right)\|F_k\| \\
&= \left(1 + \frac{L}{c} + Q\frac{L^2}{c^2}\right) m_1.
\end{aligned} \tag{7.241}$$

Where Cauchy–Schwarz inequality is used in the third inequality.
Taking $M = \left(1 + \frac{L}{c} + Q\frac{L^2}{c^2}\right) m_1$ we have (7.231).
since the sequences $\{d_k\}$ and $\{x_k\}$ are bounded, then the definition z_k in Step 4

of Algorithm 7, it holds that the sequence $\{z_k\}$ is also bounded. Therefore, by similar argument as in (7.238), there exists some positive constant, say κ, such that

$$\|F(z_k)\| \leq \kappa. \tag{7.242}$$

From (7.236), we have

$$((x_k - z_k)^T F(z_k))^2 \leq \|F(z_k)\|^2(\|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2) \tag{7.243}$$

From the line search (7.228), we have

$$\sigma^2 \alpha_k^4 \|d_k\|^4 \leq \alpha_k^2 (F(z_k)^T d_k)^2. \tag{7.244}$$

Combining (7.236) and (7.244), it holds

$$\sigma^2 \alpha_k^4 \|d_k\|^4 \leq \|F(z_k)\|^2(\|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2). \tag{7.245}$$

Since the sequence $\{\|x_k - \bar{x}\|\}$ is convergent and $\{F(z_k)\}$ is bounded, taking limit on both sides of (7.245) yields

$$\sigma^2 \lim_{k\to\infty} \alpha_k^4 \|d_k\|^4 \leq 0,$$

and hence it holds that

$$\lim_{k\to\infty} \alpha_k \|d_k\| = 0. \tag{7.246}$$

Combining (7.245) with the definition of z_k implies (7.232) holds. On the other hand, from the definition of λ_k we have

$$\begin{aligned} \|x_{k+1} - x_k\| &= \|x_k - \lambda_k F(z_k) - x_k\| \\ &= \|\lambda_k F(z_k)\| \\ &\leq \|x_k - z_k\|. \end{aligned} \tag{7.247}$$

This implies (7.233). ∎

Theorem 7.3.2 *Suppose assumption (A1)-(A3) hold and* $\{x_k\}$ *be generated by Algorithm 7. Then*

$$\liminf_{k\to\infty} \|F(x_k)\| = 0. \tag{7.248}$$

Proof For the sake of contradiction, suppose that (7.248) is not true, then there exists $n_0 > 0$ such that

$$\|F(x_k)\| \geq \delta_0 \quad \text{holds}, \quad \forall k > 0. \tag{7.249}$$

Suppose that the search direction $\|d_k\| \neq 0$ unless at the solution, then there is a constant, say δ_1

$$\|d_k\| \geq \delta_1. \tag{7.250}$$

If $\alpha_k \neq \xi$, then by the definition of $\alpha_k = (\alpha_k + \alpha_k^2\gamma_k)$, $\rho^{-1}\alpha_k$ does not satisfy (7.228) i.e.,

$$-F(x_k + \rho^{-1}\alpha_k d_k)^T d_k < \sigma\rho^{-1}\alpha_k \|d_k\|^2.$$

Now, combining with (7.227) gives

$$\begin{aligned} b\|F(x_k)\|^2 &\leq -F(x_k)^T d_k, \\ &= (F(x_k + \rho^{-1}\alpha_k d_k) - F(x_k))^T d_k - F(x_k + \rho^{-1}\alpha_k d_k)^T d_k, \\ &\leq L\rho^{-1}\alpha_k \|d_k\|^2 + \sigma\rho^{-1}\alpha_k \|d_k\|^2. \\ &= \alpha_k \|d_k\|(L+\sigma)\rho^{-1}\|d_k\|. \end{aligned} \tag{7.251}$$

Therefore, from (7.251), we have

$$\alpha_k \|d_k\| > \frac{\rho}{(L+\sigma)} \frac{b\|F_k\|^2}{\|d_k\|} \geq \frac{\rho}{(L+\sigma)} \frac{c\delta_0^2}{M}. \tag{7.252}$$

The inequality (7.252) contradicts (7.246). Therefore (7.248) holds. Hence, the proof is completed. ■

7.4 Numerical Experiments

This subsection contains some numerical results. Some numerical results are provided to demonstrate the efficacy of the proposed method (TTHZP) when compared to the following existing CG methods

- Algorithm 2.4 (IHZPM) proposed in [112].
- Algorithm 2.5 (EHZPM) proposed in [112].

The computer codes are developed in Matlab 8.3.0 (R2014a) and run on a PC with a 1.80 GHz CPU and 8 GB RAM. In this experiments, the following parameters are set in TTHZP Algorithm:

- $\xi = 1$, $\sigma = 10^{-4}$, $p = 0.99$, $q = 0$, and $\rho = 0.9$.

However, the parameters of IHZPM and EHZPM Algorithms, are taken as in [112]. For all three methods, the iteration is configured to end if any of the following conditions are met: (i) when $\|F_k\| \leq 10^{-8}$, (ii) when $\|F(z_k)\| \leq 10^{-8}$, or (iii) when the number of iterations exceeds 1000 but no x_k meeting the termination criterion is found. We ran numerical experiments using the three methods on the existing nine test problems, each with a different initial point and dimensions (n) of 1000, 10,000, and 100,000. Tables 7.37-7.41 present detailed results of the experiments conducted by displaying initial starting point "IP", number of iterations "Iter", number of function evaluation "Funeval", processing time "Time (s)", and $\|F_k\|$ is the norm of the residual at the termination point. The symbol $'-'$ represents a failure of a scheme to attain a solution after 1000 iterations or insufficient memory. The following are the initial points utilized in the experiment: $x1 = (1,1,...,1)^T$, $x2 = \left(\frac{3}{5},\frac{3}{5},...,\frac{3}{5}\right)^T$, $x3 = \left(\frac{1}{2},\frac{1}{2},...,\frac{1}{2}\right)^T$, $x4 = \left(\frac{2}{5},\frac{2}{5},...,\frac{2}{5}\right)^T$, $x5 = \left(\frac{1}{10},\frac{1}{10},...,\frac{1}{10}\right)^T$, $x6 = \left(\frac{4}{5},\frac{4}{5},...,\frac{4}{5}\right)^T$, $x7 = \left(\frac{1}{4},\frac{-1}{4},...,\frac{(-1)^n}{4}\right)^T$, $x8 = \left(\frac{1}{25},\frac{1}{25},...,\frac{1}{25}\right)^T$, $x9 = \left(\frac{1}{3},\frac{-1}{3},...,\frac{(-1)^n}{3}\right)^T$, $x10 = \left(-\frac{1}{100},-\frac{1}{100},...,-\frac{1}{100}\right)^T$. However, the initial point $x1$ is the solution of Problems 10. Therefore, this problem was solved using the remaining initial points in Table 7.41, that is, $x2 - x10$. In the experiments, the following test problems were used:

Problem 1 [13]
$F_1(x) = e^{x_1} - 1,$
$F_i(x) = e^{x_i} + x_{i-1} - 1, \quad i = 2,3,\ldots,n-1.$

Problem 2 [112]
$F_i(x) = log(x_i + 1) - \frac{x_i}{n}, \quad i = 2,3,\ldots,n.$

Problem 3 [112]
$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1,2,\ldots,n.$

Problem 4
$F_i(x) = \cos(x_i) + x_i - 1. \quad i = 1,2,\ldots,n.$

Problem 5 [112]
$F_i(x) = e^{x_i} - 1, \quad i = 1,2,\ldots,n.$

Problem 6 [112] The discretized Chandrasehar H-equation (the problem of integral equation arising in radiative heat transfer)
$F_i(x) = x_i - \left(1 - \frac{c}{2n}\sum_{j=1}^{n}\frac{\mu_i x_j}{\mu_i + \mu_j}\right)^{-1}$, i=1,2,...,n, j=1,2,...,n,
with $c \in [0,1)$ and $\mu = \frac{i-0.5}{n}$. (In our experiment we take $c = 1$).

Problem 7 [112]
$F_1 = x_1 - e^{\cos\left(\frac{x_1+x_2}{n+1}\right)},$
$F_i = x_i - e^{\cos\left(\frac{x_{i-1}+x_i+x_{i+1}}{n+1}\right)},$
$F_n = x_n - e^{\cos\left(\frac{x_{n-1}+x_n}{n+1}\right)}, \quad i = 2,3,\ldots,n-1,$
where $\Omega = \mathbb{R}^n_+$.

Problem 8 [13]
$F_i(x) = x_i - \sin|x_i - 1|, \quad i = 1,2,\ldots,n.$

Problem 9 [112]

$$F_i(x) = x_i \cos(x_i - \tfrac{1}{n}) - x_n \left(\sin x_i - 1 - (x_i - 1)^2 - \tfrac{1}{n} \sum_{i=1}^{n} x_i \right), \quad i = 1, 2, \ldots, n.$$

Problem 10 [112]

$$F_1(x) = 3x_1 - 2x_1^2 - 2x_2 + 1,$$
$$F_i(x) = 3x_i - 2x_i^2 - 2x_{i+1} + 1,$$
$$F_i(x) = 3x_n - 2x_n^2 - 2x_{n-1} + 1, \quad i = 1, 2, \ldots, n.$$

Table 7.37: Numerical results for problems 1 & 2

			TTHZP				IHZPM				EHZPM			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
1	1,000	x1	18	20	0.036761	7.25E-09	16	17	0.022648	4.54E-09	20	22	0.039549	3.91E-09
		x2	18	20	0.021058	4.69E-09	17	19	0.015737	3.34E-09	19	21	0.019494	6.52E-09
		x3	23	25	0.01702	5.41E-09	19	21	0.014624	3.95E-09	19	21	0.013438	7.03E-09
		x4	20	22	0.015559	5.71E-09	16	18	0.013967	7.39E-09	19	21	0.014209	6.84E-09
		x5	16	18	0.014585	9.67E-09	16	18	0.013136	3.47E-09	18	20	0.01365	7.62E-09
		x6	17	19	0.016183	9.95E-09	18	20	0.016541	6.91E-09	18	20	0.013137	8.69E-09
		x7	17	19	0.015243	6.3E-09	17	19	0.015253	3.3E-09	18	20	0.014263	6.51E-09
		x8	16	18	0.015401	4.67E-09	15	17	0.013126	6.33E-09	18	20	0.012781	4.07E-09
		x9	17	19	0.016007	7.87E-09	17	19	0.014944	4.5E-09	19	21	0.013411	4.07E-09
		x10	14	16	0.012724	8.36E-09	14	16	0.012692	4.65E-09	16	18	0.012463	7.12E-09
	10,000	x1	19	21	0.068664	4.21E-09	37	39	0.150035	6.87E-09	20	22	0.065675	4.37E-09
		x2	18	20	0.075766	8.03E-09	17	19	0.075087	7.52E-09	20	22	0.069855	8.94E-09
		x3	19	21	0.077525	6.09E-09	18	20	0.073019	3.83E-09	19	21	0.061777	4.81E-09
		x4	19	21	0.075728	4.45E-09	18	20	0.084838	3.14E-09	18	20	0.063899	5.85E-09
		x5	17	19	0.069937	6.03E-09	17	19	0.058019	2.91E-09	19	21	0.06157	8.46E-09
		x6	18	20	0.068096	5.73E-09	20	22	0.085963	6.25E-09	19	21	0.07183	4.51E-09
		x7	18	20	0.069688	3.7E-09	17	19	0.074412	7.76E-09	19	21	0.061944	5.54E-09
		x8	16	18	0.063916	9.54E-09	16	18	0.068354	5.71E-09	19	21	0.060545	4.53E-09
		x9	18	20	0.070141	4.56E-09	17	19	0.073624	5.35E-09	19	21	0.062038	4.89E-09
		x10	15	17	0.059065	5.18E-09	15	17	0.060293	4.22E-09	17	19	0.068867	7.93E-09
	100,000	x1	19	21	0.493843	6.86E-09	–	–	–	–	–	–	–	–
		x2	19	21	0.526418	4.77E-09	–	–	–	–	21	23	0.470643	5.41E-09
		x3	20	22	0.491625	7.7E-09	–	–	–	–	21	23	0.459705	3.85E-09
		x4	19	21	0.474267	4.91E-09	–	–	–	–	20	22	0.441913	5.67E-09
		x5	18	20	0.445452	4.05E-09	18	20	0.440851	3.75E-09	20	22	0.401647	7.81E-09
		x6	19	21	0.469903	3.34E-09	–	–	–	–	–	–	–	–
		x7	18	20	0.4556	7.29E-09	25	27	0.690836	7.58E-09	20	22	0.415227	6.46E-09
		x8	17	19	0.428587	5.65E-09	16	18	0.40087	9.98E-09	20	22	0.40307	5E-09
		x9	18	20	0.45313	9.26E-09	–	–	–	–	21	23	0.458229	4.74E-09
		x10	15	17	0.388235	9.00E-09	16	18	0.406762	3.83E-09	18	20	0.376073	8.82E-09
2	1,000	x1	6	8	0.006587	6.52E-10	6	8	0.006603	3.26E-09	21	23	0.018055	3.64E-09
		x2	5	7	0.007581	6.03E-09	6	8	0.007437	2.11E-10	20	22	0.017478	4.16E-09
		x3	5	7	0.005959	3.00E-09	6	8	0.006788	2.98E-10	19	21	0.017544	8.36E-09
		x4	5	7	0.006457	1.36E-09	6	8	0.008042	2.31E-10	19	21	0.018713	5.47E-09
		x5	4	6	0.005274	2.69E-09	4	6	0.00505	2.52E-09	16	18	0.013618	8.71E-09
		x6	6	8	0.007447	2.26E-10	6	8	0.0068	1.1E-09	20	22	0.01882	7.05E-09
		x7	5	7	0.006165	2.98E-10	8	10	0.008471	7.99E-10	19	21	0.018228	5.48E-09
		x8	4	6	0.006526	6.14E-10	4	6	0.005305	6.11E-10	15	17	0.015545	6.47E-09
		x9	6	8	0.006733	2.36E-09	8	10	0.008612	8.68E-10	19	21	0.015714	6.23E-09
		x10	4	6	0.006213	1.35E-09	5	7	0.006023	6.88E-10	16	18	0.015788	7.14E-09
	10,000	x1	6	8	0.028325	1.41E-09	–	–	–	–	–	–	–	–
		x2	6	8	0.027008	1.37E-10	–	–	–	–	20	22	0.103719	6.03E-09
		x3	5	7	0.02477	6.68E-09	–	–	–	–	19	21	0.094654	4.66E-09
		x4	5	7	0.023423	3.00E-09	6	8	0.031557	4.76E-10	20	22	0.100849	5.66E-09
		x5	4	6	0.020451	6.03E-09	5	7	0.022135	1.9E-09	17	19	0.086231	9.2E-09
		x6	6	8	0.026568	4.78E-10	–	–	–	–	–	–	–	–
		x7	6	8	0.030225	4.88E-09	10	12	0.040806	2.5E-10	20	22	0.094794	5.32E-09
		x8	4	6	0.019762	1.34E-09	5	7	0.023957	5.22E-10	16	18	0.080683	6.78E-09
		x9	6	8	0.032765	5.05E-09	10	12	0.042314	3.74E-10	20	22	0.103286	4.03E-09
		x10	4	6	0.021618	2.99E-09	7	9	0.031875	2.72E-10	17	19	0.089897	7.74E-09
	100,000	x1	6	8	0.174608	4.27E-09	–	–	–	–	–	–	–	–
		x2	6	8	0.171625	4.16E-10	–	–	–	–	–	–	–	–
		x3	6	8	0.176654	2.04E-10	–	–	–	–	–	–	–	–
		x4	5	7	0.153172	9.14E-09	–	–	–	–	–	–	–	–
		x5	5	7	0.154385	1.84E-10	5	7	0.168487	4.49E-09	17	19	0.557022	4.71E-09
		x6	6	8	0.181897	1.45E-09	–	–	–	–	–	–	–	–
		x7	7	9	0.215196	1.48E-10	8	10	0.291893	4.13E-10	20	22	0.647656	6.3E-09
		x8	4	6	0.126974	4.07E-09	5	7	0.140061	2.57E-10	17	19	0.542378	7.06E-09
		x9	7	9	0.222925	1.53E-10	8	10	0.287844	1.85E-10	21	23	0.734054	7.15E-09
		x10	4	6	0.139459	9.1E-09	8	10	0.223636	3.39E-10	18	20	0.565397	8.59E-09

Table 7.38: Numerical results for problems 3 & 4

			TTHZP				IHZPM				EHZPM			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
3	1,000	x1	6	8	0.005942	2.4E-10	7	9	0.00622	4.13E-09	20	22	0.017711	5.97E-09
		x2	5	7	0.005086	2.51E-10	7	9	0.006095	3.08E-09	20	22	0.015857	5.67E-09
		x3	5	7	0.005473	2.91E-10	7	9	0.007453	3.35E-09	20	22	0.016378	4.95E-09
		x4	5	7	0.006134	2.98E-10	7	9	0.006467	3.04E-09	20	22	0.014669	4.09E-09
		x5	5	7	0.005361	1.08E-10	5	7	0.005651	5.1E-09	18	20	0.015148	8.68E-09
		x6	5	7	0.005871	1.13E-10	7	9	0.006562	4.87E-09	20	22	0.020166	6.51E-09
		x7	12	14	0.01285	6.55E-09	21	23	0.01828	3.66E-09	18	20	0.018488	3.29E-09
		x8	4	6	0.005565	5.5E-09	4	6	0.005418	6.27E-09	18	20	0.024221	4.35E-09
		x9	12	14	0.01189	8.84E-09	21	23	0.017708	4.77E-09	18	20	0.016622	4.4E-09
		x10	10	12	0.010439	7.58E-09	17	19	0.016126	8.9E-09	15	17	0.017367	4.83E-09
	10,000	x1	6	8	0.024162	7.58E-10	7	9	0.031274	4.18E-09	21	23	0.082867	5.67E-09
		x2	5	7	0.021552	7.93E-10	7	9	0.028923	6.38E-09	20	22	0.079572	6.89E-09
		x3	5	7	0.02279	9.2E-10	7	9	0.02858	8.99E-10	21	23	0.097106	6.31E-09
		x4	5	7	0.020435	9.42E-10	7	9	0.028707	8.73E-09	20	22	0.080728	7.88E-09
		x5	5	7	0.021897	3.42E-10	6	8	0.02198	6.43E-09	19	21	0.09699	9.65E-09
		x6	5	7	0.023938	3.56E-10	7	9	0.030868	5.8E-09	21	23	0.099906	4.5E-09
		x7	13	15	0.066552	3.82E-09	22	24	0.083569	4.03E-09	18	20	0.098144	9.47E-09
		x8	5	7	0.021195	1.74E-10	5	7	0.022752	8.08E-09	19	21	0.071794	4.84E-09
		x9	13	15	0.069197	5.16E-09	22	24	0.085593	3.87E-09	19	21	0.083308	3.01E-09
		x10	11	13	0.056696	4.42E-09	19	21	0.080104	3.62E-09	16	18	0.094939	4.58E-09
	100,000	x1	6	8	0.180678	2.4E-09	–	–	–	–	–	–	–	–
		x2	5	7	0.139855	2.51E-09	–	–	–	–	21	23	0.784733	7.39E-09
		x3	5	7	0.132149	2.91E-09	–	–	–	–	19	21	0.766242	7.39E-09
		x4	5	7	0.137804	2.98E-09	9	11	0.288251	8.23E-09	22	24	0.705285	5.63E-09
		x5	5	7	0.129873	1.08E-09	6	8	0.143563	7.31E-09	20	22	0.534168	9.26E-09
		x6	5	7	0.134165	1.13E-09	–	–	–	–	–	–	–	–
		x7	14	16	0.485152	2.23E-09	22	24	0.630109	8.07E-09	19	21	0.642433	7.17E-09
		x8	5	7	0.136125	5.5E-10	6	8	0.142574	9.09E-09	20	22	0.525555	5.34E-09
		x9	14	16	0.476908	3.01E-09	–	–	–	–	–	–	–	–
		x10	12	14	0.409385	2.58E-09	20	22	0.560778	4.1E-09	17	19	0.567985	4.35E-09
4	1,000	x1	6	8	0.005619	2.87E-09	7	9	0.006823	1.19E-10	20	22	0.021271	4.63E-09
		x2	5	7	0.00595	9.89E-09	6	8	0.005469	5.21E-10	20	22	0.018736	5.87E-09
		x3	5	7	0.005407	3.92E-09	6	8	0.005786	3.94E-10	20	22	0.021209	3.86E-09
		x4	5	7	0.005703	1.46E-09	6	8	0.005812	2.31E-10	19	21	0.017197	6.7E-09
		x5	4	6	0.005901	1.98E-09	4	6	0.00612	1.85E-09	16	18	0.016049	8.84E-09
		x6	6	8	0.007933	5.59E-10	7	9	0.006046	1.54E-10	18	20	0.017072	9.81E-09
		x7	6	8	0.00682	1.52E-09	10	12	0.008768	4.15E-09	19	21	0.019414	5.55E-09
		x8	4	6	0.006465	4.2E-10	4	6	0.004756	4.18E-10	15	17	0.013598	6.25E-09
		x9	6	8	0.006017	1.58E-09	10	12	0.007923	4.82E-09	19	21	0.018768	6.64E-09
		x10	4	6	0.005274	9.07E-10	7	9	0.006779	3.38E-09	16	18	0.014634	6.93E-09
	10,000	x1	6	8	0.025622	9.09E-09	8	10	0.033583	2.22E-10	21	23	0.089595	8.04E-09
		x2	6	8	0.022932	3.13E-10	6	8	0.024428	2.93E-09	21	23	0.084526	6.5E-09
		x3	6	8	0.023912	1.24E-10	6	8	0.024945	2.8E-09	20	22	0.085129	6.81E-09
		x4	5	7	0.020152	4.63E-09	6	8	0.024826	5E-09	17	19	0.071388	4.43E-09
		x5	4	6	0.017415	6.25E-09	5	7	0.02025	1.95E-09	17	19	0.07388	9.76E-09
		x6	6	8	0.02415	1.77E-09	7	9	0.031434	1.05E-10	20	22	0.08945	5.43E-09
		x7	6	8	0.025838	4.79E-09	11	13	0.037641	1.13E-10	20	22	0.080899	5.62E-09
		x8	4	6	0.017359	1.33E-09	5	7	0.020348	5.16E-10	16	18	0.060966	6.95E-09
		x9	6	8	0.026715	5.01E-09	10	12	0.034902	1.05E-10	20	22	0.074722	4.89E-09
		x10	4	6	0.01742	2.87E-09	8	10	0.029054	1.07E-10	17	19	0.066232	7.72E-09
	100,000	x1	7	9	0.174274	2.87E-10	–	–	–	–	–	–	–	–
		x2	6	8	0.15497	9.89E-10	–	–	–	–	21	23	0.593699	3.56E-09
		x3	6	8	0.156447	3.92E-10	–	–	–	–	21	23	0.576189	4.95E-09
		x4	6	8	0.160227	1.46E-10	–	–	–	–	21	23	0.554794	7.76E-09
		x5	5	7	0.129201	1.98E-10	5	7	0.1435	4.93E-09	17	19	0.436677	5.36E-09
		x6	6	8	0.182174	5.59E-09	–	–	–	–	–	–	–	–
		x7	7	9	0.185465	1.52E-10	10	12	0.269967	2.61E-10	20	22	0.502666	9.89E-09
		x8	4	6	0.117002	4.2E-09	5	7	0.117484	2.69E-10	17	19	0.458408	7.29E-09
		x9	7	9	0.186866	1.58E-10	6	8	0.20446	8.05E-09	19	21	0.533535	4.36E-09
		x10	4	6	0.117006	9.07E-09	8	10	0.181251	3.38E-10	18	20	0.471752	8.59E-09

Table 7.39: Numerical results for problems 5 & 6

			TTHZP				IHZPM				EHZPM			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
5	1,000	x1	7	9	0.00676	1.54E-09	11	13	0.007697	6.09E-09	20	22	0.015367	3.91E-09
		x2	6	8	0.006472	1.5E-09	10	12	0.008317	5.67E-09	19	21	0.015108	6.52E-09
		x3	6	8	0.006201	1.57E-09	10	12	0.008339	3.42E-09	19	21	0.014058	7.03E-09
		x4	6	8	0.005654	1.58E-09	10	12	0.007987	4.6E-09	19	21	0.015544	6.84E-09
		x5	5	7	0.006229	1.57E-09	9	11	0.007522	4.95E-09	18	20	0.014218	7.62E-09
		x6	4	6	0.004118	8.92E-10	9	11	0.00724	2.82E-09	18	20	0.014127	8.69E-09
		x7	5	7	0.005704	2.27E-10	5	7	0.005422	5.77E-09	18	20	0.012952	6.51E-09
		x8	4	6	0.005326	8.95E-10	9	11	0.007176	2.65E-09	18	20	0.012761	4.07E-09
		x9	5	7	0.006035	5.87E-10	6	8	0.006561	1.19E-10	19	21	0.014354	4.07E-09
		x10	3	5	0.004241	1.58E-09	4	6	0.004547	1.53E-09	16	18	0.012745	7.12E-09
	10,000	x1	7	9	0.027813	4.86E-09	12	14	0.039537	1.7E-10	20	22	0.069167	4.37E-09
		x2	6	8	0.022781	4.74E-09	11	13	0.037809	1.57E-10	20	22	0.064303	8.94E-09
		x3	6	8	0.021258	4.97E-09	11	13	0.033413	1.3E-10	19	21	0.062573	4.81E-09
		x4	6	8	0.02195	4.99E-09	11	13	0.034558	1.27E-10	18	20	0.059762	5.85E-09
		x5	5	7	0.018217	4.98E-09	10	12	0.030144	1.53E-10	19	21	0.06161	8.46E-09
		x6	4	6	0.014701	2.82E-09	6	8	0.025164	2.84E-09	19	21	0.068317	4.51E-09
		x7	5	7	0.017739	7.19E-10	6	8	0.020381	1.12E-10	19	21	0.061574	5.54E-09
		x8	4	6	0.015018	2.83E-09	9	11	0.028777	8.37E-09	19	21	0.065709	4.53E-09
		x9	5	7	0.01815	1.86E-09	6	8	0.020543	1.25E-09	19	21	0.076442	4.89E-09
		x10	3	5	0.01215	5E-09	4	6	0.015282	4.84E-09	17	19	0.061914	7.93E-09
	100,000	x1	8	10	0.172356	1.54E-10	–	–	–	–	–	–	–	–
		x2	7	9	0.147858	1.5E-10	–	–	–	–	21	23	0.476847	5.41E-09
		x3	7	9	0.151455	1.57E-10	–	–	–	–	21	23	0.500895	3.85E-09
		x4	7	9	0.165042	1.58E-10	–	–	–	–	20	22	0.447635	5.67E-09
		x5	6	8	0.133655	1.57E-10	8	10	0.15607	3.36E-10	20	22	0.399647	7.81E-09
		x6	4	6	0.093881	8.92E-09	–	–	–	–	–	–	–	–
		x7	5	7	0.109231	2.27E-09	6	8	0.145751	5.3E-09	20	22	0.473855	6.46E-09
		x8	4	6	0.100766	8.95E-09	9	11	0.161947	5.6E-10	20	22	0.406266	5E-09
		x9	5	7	0.107399	5.87E-09	10	12	0.238174	5.39E-10	21	23	0.450178	4.74E-09
		x10	4	6	0.08967	1.58E-10	5	7	0.092156	1.51E-10	18	20	0.390821	8.82E-09
6	1,000	x1	8	10	0.012043	9.74E-09	17	19	0.019679	8.73E-09	16	18	0.021584	4.96E-09
		x2	12	14	0.016164	4.6E-09	18	20	0.020792	9.61E-09	16	18	0.019559	5.33E-09
		x3	12	14	0.015438	2.09E-09	18	20	0.019869	8.83E-09	14	16	0.017468	3.33E-09
		x4	11	13	0.01453	3.88E-09	18	20	0.021325	3.8E-09	14	16	0.019084	3.83E-09
		x5	11	13	0.019421	2.14E-09	17	19	0.019787	9.85E-09	14	16	0.018743	4.62E-09
		x6	13	15	0.016606	2.29E-09	17	19	0.022128	3.74E-09	16	18	0.021872	3.86E-09
		x7	11	13	0.016568	8.52E-09	19	21	0.022795	2.91E-09	15	17	0.019811	2.41E-09
		x8	9	11	0.011429	3.3E-09	17	19	0.020545	5.81E-09	12	14	0.017294	4.21E-09
		x9	10	12	0.013478	4.07E-09	16	18	0.018605	7.85E-09	15	17	0.014079	5.09E-09
		x10	7	9	0.009764	9.74E-09	14	16	0.017425	2.99E-09	12	14	0.013235	4.81E-09
	10,000	x1	10	12	0.068632	5.02E-09	15	17	0.096181	5.21E-09	17	19	0.117131	6.27E-09
		x2	11	13	0.066346	2.46E-09	15	17	0.088532	3.35E-09	17	19	0.11579	9.41E-09
		x3	10	12	0.062165	6.81E-09	13	15	0.074248	3.21E-09	15	17	0.105101	9.16E-09
		x4	10	12	0.068928	2.04E-09	14	16	0.079345	2.98E-09	15	17	0.098544	7.04E-09
		x5	9	11	0.062373	6.68E-09	15	17	0.085929	5.87E-09	14	16	0.092003	9.78E-09
		x6	11	13	0.077969	7.51E-09	14	16	0.090732	7.75E-09	15	17	0.100506	9.25E-09
		x7	10	12	0.061319	4.53E-09	15	17	0.084891	6.17E-09	16	18	0.109527	8.97E-09
		x8	7	9	0.059258	6.88E-09	15	17	0.09133	3.93E-09	15	17	0.102201	9.48E-09
		x9	9	11	0.057741	2E-09	14	16	0.083351	3.78E-09	15	17	0.099291	9.57E-09
		x10	9	11	0.061101	5.02E-09	10	12	0.066903	6.41E-09	13	15	0.102688	9.08E-09
	100,000	x1	9	11	0.380941	2.06E-09	–	–	–	–	–	–	–	–
		x2	9	11	0.387209	7.41E-09	–	–	–	–	17	19	0.980548	2.54E-09
		x3	9	11	0.393752	3.28E-09	–	–	–	–	17	19	0.888177	2.96E-09
		x4	8	10	0.37006	5.54E-09	15	17	0.692001	7.26E-09	15	17	0.878389	2.29E-09
		x5	8	10	0.36469	1.99E-09	16	18	0.620307	3.15E-09	18	20	0.883358	3.6E-09
		x6	10	12	0.488956	3.77E-09	–	–	–	–	–	–	–	–
		x7	9	11	0.453113	2.13E-09	14	16	0.594301	7.05E-09	16	18	0.81286	6.73E-09
		x8	10	12	0.490746	9.51E-09	12	14	0.47552	4.93E-09	17	19	0.842603	3.75E-09
		x9	7	9	0.317164	2.53E-09	15	17	0.676554	5.94E-09	18	20	1.074665	3.68E-09
		x10	8	10	0.362125	2.06E-09	15	17	0.607184	6.93E-09	14	16	0.786929	7.73E-09

Table 7.40: Numerical results for problems 7 & 8

			TTHZP				IHZPM				EHZPM			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
7	1,000	x1	8	10	0.010557	2.91E-09	9	11	0.01233	5.05E-10	21	23	0.025551	6.53E-09
		x2	8	10	0.010211	3.69E-09	10	12	0.014542	4.39E-09	19	21	0.027144	4.95E-09
		x3	8	10	0.009844	3.89E-09	11	13	0.01505	1.84E-10	21	23	0.023852	7.66E-09
		x4	8	10	0.011307	4.09E-09	11	13	0.013188	3.71E-10	21	23	0.029021	5.6E-09
		x5	8	10	0.009348	4.72E-09	12	14	0.015335	4.02E-10	21	23	0.025973	8.92E-09
		x6	8	10	0.009322	3.29E-09	11	13	0.013053	2.56E-10	21	23	0.025945	4.2E-09
		x7	8	10	0.008962	5.48E-09	12	14	0.016211	1.53E-10	20	22	0.027174	6.02E-09
		x8	8	10	0.010708	4.83E-09	12	14	0.014927	1.88E-10	21	23	0.026292	7.59E-09
		x9	8	10	0.009526	5.67E-09	12	14	0.013969	2.21E-10	21	23	0.023062	8.7E-09
		x10	8	10	0.009539	4.96E-09	12	14	0.015007	2.64E-10	21	23	0.024413	5.78E-09
	10,000	x1	7	9	0.040889	6.7E-10	–	–	–	–	21	23	0.127687	4.23E-09
		x2	7	9	0.040382	8.54E-10	–	–	–	–	20	22	0.128844	6.96E-09
		x3	7	9	0.038999	9.02E-10	–	–	–	–	22	24	0.150906	3.76E-09
		x4	7	9	0.038577	9.5E-10	–	–	–	–	21	23	0.183199	5.79E-09
		x5	7	9	0.037446	1.1E-09	–	–	–	–	–	–	–	–
		x6	7	9	0.038935	7.61E-10	–	–	–	–	22	24	0.130399	8.58E-09
		x7	7	9	0.040569	1.28E-09	–	–	–	–	–	–	–	–
		x8	7	9	0.054202	1.13E-09	–	–	–	–	–	–	–	–
		x9	7	9	0.036501	1.33E-09	–	–	–	–	–	–	–	–
		x10	7	9	0.037553	1.16E-09	–	–	–	–	–	–	–	–
	100,000	x1	7	9	0.320677	6.73E-10	–	–	–	–	–	–	–	–
		x2	7	9	0.275229	8.58E-10	–	–	–	–	–	–	–	–
		x3	7	9	0.323518	9.06E-10	–	–	–	–	–	–	–	–
		x4	7	9	0.271228	9.55E-10	–	–	–	–	–	–	–	–
		x5	7	9	0.26768	1.1E-09	–	–	–	–	–	–	–	–
		x6	7	9	0.267562	7.64E-10	–	–	–	–	–	–	–	–
		x7	7	9	0.25155	1.29E-09	–	–	–	–	–	–	–	–
		x8	7	9	0.258587	1.13E-09	–	–	–	–	–	–	–	–
		x9	7	9	0.261322	1.33E-09	–	–	–	–	–	–	–	–
		x10	7	9	0.250103	1.16E-09	–	–	–	–	–	–	–	–
8	1,000	x1	14	16	0.013145	2.53E-09	19	21	0.015142	4.51E-09	16	18	0.015983	7.33E-09
		x2	13	15	0.011644	3.02E-09	18	20	0.015376	5.13E-09	15	17	0.014381	6.78E-09
		x3	11	13	0.010263	6.73E-09	16	18	0.013695	4.76E-09	13	15	0.01355	9.43E-09
		x4	13	15	0.011276	2.74E-09	18	20	0.015435	4.45E-09	15	17	0.012986	5.92E-09
		x5	14	16	0.012295	3.22E-09	19	21	0.014737	6.93E-09	16	18	0.015932	8.16E-09
		x6	13	15	0.012982	7.65E-09	19	21	0.014835	4.28E-09	16	18	0.015475	4.81E-09
		x7	14	16	0.013548	8.16E-09	16	18	0.012236	8.85E-09	14	16	0.014473	8.64E-09
		x8	14	16	0.012695	3.78E-09	19	21	0.016126	7.68E-09	16	18	0.015562	9.45E-09
		x9	14	16	0.011986	9.77E-09	18	20	0.016402	5.46E-09	15	17	0.014401	3.53E-09
		x10	14	16	0.013313	4.51E-09	19	21	0.020132	8.35E-09	17	19	0.01597	3.01E-09
	10,000	x1	14	16	0.058877	8.01E-09	17	19	0.071658	4.69E-09	17	19	0.085409	3.39E-09
		x2	13	15	0.053878	9.55E-09	19	21	0.068069	5.03E-09	16	18	0.086833	5.8E-09
		x3	12	14	0.055133	4.62E-09	17	19	0.064957	5.01E-09	14	16	0.075968	8.11E-09
		x4	13	15	0.05626	8.66E-09	19	21	0.073393	4.57E-09	16	18	0.087378	5.08E-09
		x5	15	17	0.068114	2.21E-09	20	22	0.073496	6.38E-09	17	19	0.09017	3.34E-09
		x6	14	16	0.055157	5.25E-09	19	21	0.075769	8.97E-09	17	19	0.09626	2.73E-09
		x7	15	17	0.066863	5.6E-09	19	21	0.073209	7.47E-09	15	17	0.078666	4.64E-09
		x8	15	17	0.056879	2.59E-09	19	21	0.075977	4E-09	16	18	0.082456	5.99E-09
		x9	15	17	0.061671	6.7E-09	20	22	0.08161	7.27E-09	16	18	0.095701	6.35E-09
		x10	15	17	0.06033	3.09E-09	20	22	0.104739	4.2E-09	18	20	0.097309	3.17E-09
	100,000	x1	15	17	0.398271	5.5E-09	–	–	–	–	18	20	0.688944	5.49E-09
		x2	14	16	0.380177	6.55E-09	20	22	0.666958	6.09E-09	16	18	0.577961	9.14E-09
		x3	13	15	0.345049	3.17E-09	18	20	0.657546	5.26E-09	15	17	0.534358	6.98E-09
		x4	14	16	0.378672	5.94E-09	19	21	0.610211	4.12E-09	17	19	0.62249	3.57E-09
		x5	15	17	0.400157	6.99E-09	–	–	–	–	18	20	0.664593	6.8E-09
		x6	15	17	0.388646	3.6E-09	20	22	0.571283	7.15E-09	17	19	0.76903	7.4E-09
		x7	16	18	0.420976	3.84E-09	–	–	–	–	19	21	0.768806	6.77E-09
		x8	15	17	0.385743	8.2E-09	–	–	–	–	18	20	0.700174	7.41E-09
		x9	16	18	0.416979	4.6E-09	–	–	–	–	–	–	–	–
		x10	15	17	0.54679	9.77E-09	–	–	–	–	18	20	0.725455	3.82E-09

Table 7.41: Numerical results for problems 9 & 10

			TTHZP				IHZPM				EHZPM			
Problems	Dimension	IP	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$	Iter	Funeval	Time(s)	$\|F_k\|$
9	1,000	x1	14	16	0.020469	3.38E-09	22	24	0.03124	5.1E-09	17	19	0.031742	5.11E-09
		x2	13	15	0.023402	3.18E-09	21	23	0.031896	4.34E-09	18	20	0.030163	8.28E-09
		x3	12	14	0.021335	7.92E-09	21	23	0.029192	3.78E-09	18	20	0.033684	5.24E-09
		x4	12	14	0.018372	1.94E-09	20	22	0.028301	7.45E-09	17	19	0.02955	9.93E-09
		x5	12	14	0.019083	3.64E-09	20	22	0.025039	4.94E-09	17	19	0.027181	5.31E-09
		x6	13	15	0.019934	9.05E-09	20	22	0.027777	5.22E-09	19	21	0.028153	5.11E-09
		x7	11	13	0.017712	5.97E-09	20	22	0.028376	6.04E-09	17	19	0.024195	6.41E-09
		x8	11	13	0.018268	8.36E-09	19	21	0.024042	6.28E-09	16	18	0.025918	8.01E-09
		x9	13	15	0.018318	1.9E-09	20	22	0.026954	6.09E-09	17	19	0.02582	6.95E-09
		x10	10	12	0.016309	7.3E-09	17	19	0.024152	8.74E-09	15	17	0.025734	4.73E-09
	10,000	x1	15	17	0.116284	1.97E-09	21	23	0.182182	9.75E-09	20	22	0.192527	4.8E-09
		x2	14	16	0.10961	1.86E-09	22	24	0.155455	7.99E-09	19	21	0.192352	6.74E-09
		x3	13	15	0.100289	4.63E-09	22	24	0.191745	5.27E-09	18	20	0.167451	5.77E-09
		x4	12	14	0.111041	6.15E-09	22	24	0.153829	3.69E-09	18	20	0.175169	9.66E-09
		x5	13	15	0.165186	2.12E-09	21	23	0.149497	5.47E-09	18	20	0.172095	5.03E-09
		x6	14	16	0.100772	5.29E-09	22	24	0.164789	4.73E-09	20	22	0.190027	3.68E-09
		x7	12	14	0.096057	3.48E-09	20	22	0.146935	3.96E-09	18	20	0.172834	4.93E-09
		x8	12	14	0.108705	4.88E-09	20	22	0.169326	7.11E-09	17	19	0.167154	7.6E-09
		x9	13	15	0.118699	6.01E-09	20	22	0.145191	6.89E-09	17	19	0.1808	5.15E-09
		x10	11	13	0.083155	4.25E-09	18	20	0.13145	9.9E-09	16	18	0.158308	4.49E-09
	100,000	x1	15	17	0.91504	6.24E-09	–	–	–	–	–	–	–	–
		x2	14	16	0.867538	5.88E-09	–	–	–	–	–	–	–	–
		x3	14	16	0.765665	2.7E-09	–	–	–	–	20	22	1.654807	4.88E-09
		x4	13	15	0.751449	3.59E-09	–	–	–	–	18	20	1.346615	4.4E-09
		x5	13	15	0.729771	6.72E-09	21	23	1.106345	3.9E-09	19	21	1.453386	3.97E-09
		x6	15	17	0.823708	3.08E-09	–	–	–	–	–	–	–	–
		x7	13	15	0.697483	2.03E-09	23	25	1.602479	4.92E-09	19	21	1.407026	4.31E-09
		x8	13	15	0.701442	2.84E-09	21	23	1.197109	6.66E-09	18	20	1.352549	7.13E-09
		x9	14	16	0.755143	3.5E-09	–	–	–	–	19	21	1.375671	8.81E-09
		x10	12	14	0.665132	2.48E-09	20	22	1.033497	4.02E-09	17	19	1.360813	4.26E-09
10	1,000	x2	14	16	0.014524	2.28E-09	21	23	0.023109	4.45E-09	18	20	0.021997	6.12E-09
		x3	13	15	0.015234	2.04E-09	22	24	0.022636	7.02E-09	18	20	0.021251	4.32E-09
		x4	14	16	0.014873	1.91E-09	20	22	0.019614	9.01E-09	19	21	0.020221	4.12E-09
		x5	13	15	0.013688	6.31E-09	21	23	0.02173	8.92E-09	19	21	0.024237	3.68E-09
		x6	14	16	0.013282	1.89E-09	21	23	0.019163	7.56E-09	17	19	0.021155	5.78E-09
		x7	13	15	0.014566	2.77E-09	19	21	0.02147	8.13E-09	17	19	0.020174	3.26E-09
		x8	13	15	0.014625	4.24E-09	21	23	0.019272	7.12E-09	18	20	0.023302	9.29E-09
		x9	12	14	0.017596	7.64E-09	18	20	0.020901	6.05E-09	15	17	0.018886	9.28E-09
		x10	13	15	0.015029	2.49E-09	21	23	0.022668	5.4E-09	18	20	0.02154	6.55E-09
	10,000	x2	14	16	0.113189	7.2E-09	–	–	–	–	17	19	0.1052	9.63E-09
		x3	13	15	0.072997	6.45E-09	–	–	–	–	18	20	0.107023	9.26E-09
		x4	14	16	0.085927	6.04E-09	–	–	–	–	19	21	0.118409	4.78E-09
		x5	14	16	0.060437	3.68E-09	22	24	0.151209	6.34E-09	19	21	0.118739	4.35E-09
		x6	14	16	0.072375	5.96E-09	21	23	0.108386	4.57E-09	20	22	0.115235	5.7E-09
		x7	13	15	0.063288	8.77E-09	20	22	0.093252	4.79E-09	17	19	0.106091	7.05E-09
		x8	14	16	0.077851	2.47E-09	21	23	0.090118	3.97E-09	19	21	0.114839	4.58E-09
		x9	13	15	0.073467	4.45E-09	19	21	0.09069	6.97E-09	16	18	0.095777	8.07E-09
		x10	13	15	0.061811	7.86E-09	21	23	0.087473	8.79E-09	19	21	0.115358	4.63E-09
	100,000	x2	15	17	0.555942	4.2E-09	–	–	–	–	–	–	–	–
		x3	14	16	0.558755	3.76E-09	–	–	–	–	–	–	–	–
		x4	15	17	0.560562	3.52E-09	–	–	–	–	–	–	–	–
		x5	15	17	0.583609	2.14E-09	–	–	–	–	–	–	–	–
		x6	15	17	0.555493	3.48E-09	–	–	–	–	–	–	–	–
		x7	14	16	0.530348	5.11E-09	22	24	0.951946	8.43E-09	18	20	0.978537	5.75E-09
		x8	14	16	0.53403	7.82E-09	–	–	–	–	–	–	–	–
		x9	14	16	0.543142	2.6E-09	20	22	0.956008	6.07E-09	17	19	0.934911	8.32E-09
		x10	14	16	0.543442	4.58E-09	–	–	–	–	–	–	–	–

A careful analysis of Tables 7.37-7.41 reveals that the TTHZP method outperforms the IHZPM and EHZPM methods since the former solved all of the problems in the experiments and the latter failed to solve some of the problems during the iteration process by clear indication from the tables. Furthermore, Because it has fewer iterations, the TTHZP method converges to the solution of (1.1) faster than the IHZPM and EHZPM methods. Indeed, the proposed method outperforms the compared methods since it takes less processing time, iterations, and function evaluation.

To have an insight in to the results displayed, the data presented in Tables 7.37-7.41 was analyzed in Figures 7.23-7.25 using the Dolan and Mor*é* performance profile [44]. Figures 7.23-7.25 show the performance of all three methods in terms of iterations, function evaluations, and processing time.A fraction p of the problems for which a method is within a factor τ of the best time is plotted for each of the three methods. The vertical axis in Figures 7.23-7.25 clearly shows that the TTHZP method successfully solved a considerably larger percentage of the problems than the IHZPM and EHZPM methods in terms of the three metrics stated above. Furthermore, the top curves in all three figures correspond to the scheme with the best performance, which is clearly the TTHZP approach. Figures 7.23 and 7.24 show that the TTHZP method perform effectively and solves around 76% of the problems when compared to the existing methods. Also, as shown in Figure 7.25, the TTHZP method performs well and solves approximately 87% of the problems. In light of the foregoing, we conclude that the TTHZP method is promising and appropriate for solving large-scale monotone nonlinear equations.

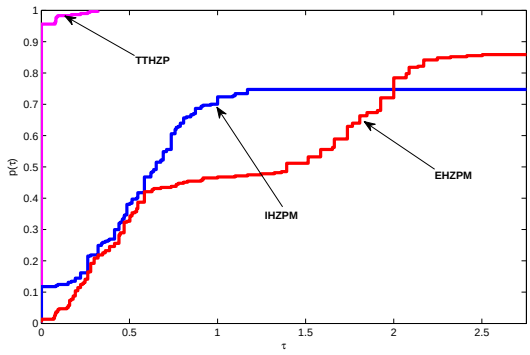

Figure 7.23: Performance profile for the number of iterations.

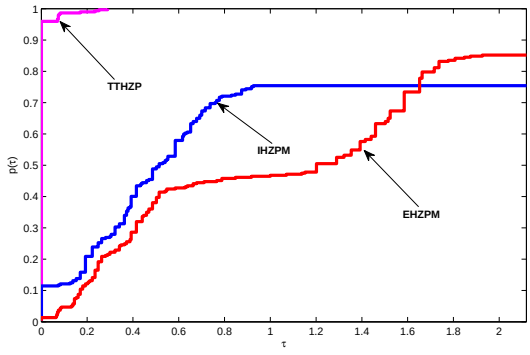

Figure 7.24: Performance profile for the function evaluation.

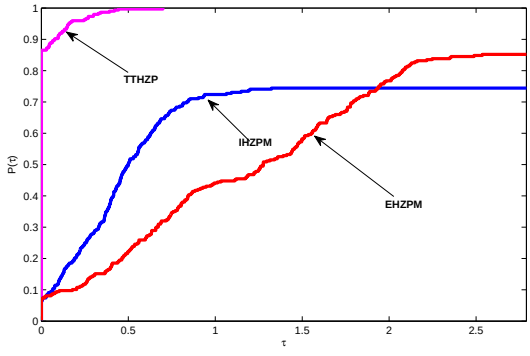

Figure 7.25: Performance profile for the CPU time (in seconds).

7.5 Conclusion

This chapter presented a three-term Hager-Zhang method established for solving monotone nonlinear equations. The global convergence analysis of the proposed algorithm was established using basic assumptions. Numerical comparisons using large-scale test problems convincingly demonstrated the efficiency of the TTHZ approach, which surpassed the IHZPM and EHZPM methods [112], as shown in Tables 7.37-7.41. Furthermore, The numerical data reported in Tables 7.37-7.41 were examined in Figures 7.23-7.25 using the Dolan and Mor*é* performance profile [44], demonstrating the efficiency of the proposed method.

CHAPTER EIGHT

On the Modified Dai-Yuan Projection Method for Monotone Nonlinear Equations with Convex Constraint and its Application

9.1 Introduction

This paper presents a method for constrained system of monotone nonlinear equations, y combining a modified Dai-Yuan (DY) search direction with the projection scheme. The new scheme can be seen as an adaptation of the iterative method designed by Yu and Guan (2005) for unconstrained optimization. Some nice properties of the scheme include being derivative-free and satisfying the required condition for global convergence irrespective of the linesearch strategy employed. Moreover, under mild conditions, the proposed method converges globally. Numerical experiments conducted with some recent variants of the DY method show that the proposed method is quite effective.

In recent decades, researchers working on large-dimension problems have focused on conjugate gradient (CG) methods, which, compared to other methods, are not only derivative-free but also require low memory to implement [35, 50, 66, 88]. These methods were mostly developed to solve the following unconstrained optimization problem

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be continuous, bounded below and its gradient denoted by $\nabla f(x_k)$ exists. Typically, iterative procedure of the aforementioned

methods are established by

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, \ldots,$$

where, $s_k = x_{k+1} - x_k$, $\alpha_k > 0$ is a step length that can be computed using any suitable line search, and the search directions. The CG search direction for unconstrained optimization problem is given by

$$d_0 = \nabla f(x_0), \quad d_k = -\nabla f(x_k) + \beta_k d_{k-1}, \quad k = 1, 2, ..., \tag{9.253}$$

where β_k is a scalar specifying the CG methods.

Although some of these methods are computationally efficient with strong convergence properties [66, 88], they, however, fail to satisfy the following sufficient descent condition.

$$\nabla f(x_k)^T d_k < -\tau \|\nabla f(x_k)\|^2, \quad \forall k, \quad \tau > 0, \tag{9.254}$$

9.2 Motivation and the Proposed Method

The method proposed by Dai and Yuan (DY) in [35] always generates descent directions when using a standard Wolfe line search and thus converges globally when the Lipschitz assumption holds. However, one disadvantage of the DY method is that its numerical results are unsatisfactory.

Now, keep in mind that the sufficient descent condition provided in (9.254) is an essential factor in global convergence of the CG methods. In [126], Yu and Guan proposed the following modified version of the DY parameter

$$\beta_k^{DDY} = \beta_k^{DY} - \frac{C\|\nabla f(x_k)\|^2}{(d_{k-1}^T y_{k-1})^2} d_{k-1}^T \nabla f(x_k), \tag{9.255}$$

where the sufficient descent condition (9.254) holds for $\tau = 1 - \frac{1}{4C}$ with $C \geq \frac{1}{4}$. Inspired by the approach [126], Yu et al. [127] presented the spectral DY version of the scheme in (9.255) with search direction as follows:

$$d_k = -\frac{1}{\delta_k}\nabla f(x_k) + \beta_k^{DSDY} d_{k-1}, \tag{9.256}$$

with

$$\beta_k^{DSDY} = \frac{\|\nabla f(x_k)\|^2}{\delta_k d_{k-1}^T y_{k-1}} - \frac{C\|\nabla f(x_k)\|^2}{\delta_k (d_{k-1}^T y_{k-1})^2} d_{k-1}^T \nabla f(x_k), \quad C \geq \frac{1}{4}, \tag{9.257}$$

where $\delta_k = \frac{y_{k-1}^T s_{k-1}}{s_{k-1}^T s_{k-1}}$.

Despite the fact that the methods presented in [126], and [127], satisfied the sufficient descent condition for $C \geq \frac{1}{4}$, they do not necessarily converge globally for $C \in (0, \frac{1}{4})$. As a result, we intend to propose a DY type method that satisfies (9.254) for $C \in (0, \infty)$. Therefore, the parameter β_k^{DDY} in (9.255) can be updated as follows:

$$\beta_k^{UDDY} = \zeta \beta_k^{DY} - \zeta \frac{C_k \|\nabla f(x_k)\|^2}{(d_{k-1}^T y_{k-1})^2} d_{k-1}^T \nabla f(x_k), \tag{9.258}$$

where $\zeta > 0$. Using (9.253) and (9.258) the search direction can be written as

$$d_k = -\nabla f(x_k) + \zeta \frac{\|\nabla f(x_k)\|^2}{d_{k-1}^T y_{k-1}} s_{k-1} - \zeta \frac{C_k \|\nabla f(x_k)\|^2 d_{k-1}^T \nabla f(x_k)}{(d_{k-1}^T y_{k-1})^2} s_{k-1}. \tag{9.259}$$

Based on Perry's idea [83], the proposed approach's search direction can be written as

$$d_k = -H_k \nabla f(x_k), \tag{9.260}$$

where H_k denotes the search direction matrix, which can be expressed as

$$H_k = I - \zeta \frac{s_{k-1} \nabla f(x_k)^T}{y_{k-1}^T s_{k-1}} + \zeta \frac{C_k \|\nabla f(x_k)\|^2 s_{k-1} s_{k-1}^T}{(y_{k-1}^T s_{k-1})^2}. \tag{9.261}$$

where I is an identity matrix. It's clear that H_k isn't symmetric. It can, however, be symmetrized as

$$Q_k = I - \zeta \frac{s_{k-1} \nabla f(x_k)^T + \nabla f(x_k) s_{k-1}^T}{y_{k-1}^T s_{k-1}} + \zeta \frac{C_k \|\nabla f(x_k)\|^2 s_{k-1} s_{k-1}^T}{(y_{k-1}^T s_{k-1})^2}. \tag{9.262}$$

As a result, the scheme's newly revised search direction is

$$d_k = -Q_k \nabla f(x_k), \tag{9.263}$$

where Q_k is given in (9.262).

Theorem 9.2.1 *Let the symmetric matrix Q_k be defined in (9.262). Then, its eigenvalues consists $1(n-2)$ multiplicity, η_k^+ and η_k^-, where*

$$\eta_k^+ = \frac{1}{2}\left[(2-2\zeta p_k+\zeta C_k q_k)+\sqrt{(\zeta C_k q_k-2\zeta p_k)^2+4\zeta^2(q_k-p_k^2)}\right] \tag{9.264}$$

$$\eta_k^- = \frac{1}{2}\left[(2-2\zeta p_k+\zeta C_k q_k)-\sqrt{(\zeta C_k q_k-2\zeta p_k)^2+4\zeta^2(q_k-p_k^2)}\right] \tag{9.265}$$

with

$$p_k = \frac{\nabla f(x_k)^T s_{k-1}}{y_{k-1}^T s_{k-1}}, \qquad q_k = \frac{\|\nabla f(x_k)\|^2 \|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2}. \tag{9.266}$$

Proof Let us begin by defining the Wolfe line search conditions as follows

$$f(x_k+\alpha_k d_k)-f(x_k) \le \gamma \alpha_k \nabla f(x_k)^T d_k. \tag{9.267}$$

$$\nabla f(x_k+\alpha_k d_k)^T d_k \ge \phi \nabla f(x_k)^T d_k, \tag{9.268}$$

where, $0<\gamma<\phi$. Sun and Yuan [99] also ensured that, under the Wolfe line search conditions, $y_{k-1}^T s_{k-1} > 0$. which implies that y_{k-1} and s_{k-1} are nonzero vectors. Also $\nabla f(x_k)$ is a non zero vector provided the solution is not attained. Let the span $\{\nabla f(x_k), s_{k-1}\} = V \subset \mathbb{R}^n$. Then $\dim(V) \le 2$ and $\dim(V^\perp) \ge n-2$, where $V^\perp$ represents orthogonal complement of V. However, there exists a set of mutually orthogonal vectors $\{\phi_{k-1}^i\}_{i=1}^{n-2} \subset V^\perp$ such that

$$y_{k-1}^T \phi_{k-1}^i = s_{k-1}^T \phi_{k-1}^i = 0, \;\; i = 1,2,\cdots n-2. \tag{9.269}$$

Applying (9.269) to (9.262), leads to

$$Q_k \phi_{k-1}^i = \phi_{k-1}^i, \quad i = 1,2,\cdots n-2. \tag{9.270}$$

Accordingly, ϕ_{k-1}^{i} for $i = 1,2,\cdots n-2$ are the eigenvectors of Q_k, each with an eigenvalue of 1. Then we want to find the remaining two eigenvalues. Let the two remaining eigenvalues be η_k^+ and η_k^-, respectively.

We can now present Q_k as the rank-two, as update shown below.

$$Q_k = I - \zeta \frac{s_{k-1}\nabla f(x_k)^T}{y_{k-1}^T s_{k-1}} + \zeta \left(\frac{C_k \|\nabla f(x_k)\|^2 s_{k-1} - (y_{k-1}^T s_{k-1})\nabla f(x_k)}{(y_{k-1}^T s_{k-1})^2} \right) s_{k-1}^T. \tag{9.271}$$

Using the fundamental formula of algebra [99],

$$\det(I + a_1 a_2^T + a_3 a_4^T) = (1 + a_1^T a_2)(1 + a_3^T a_4) - (a_1^T a_4)(a_2^T a_3),$$

where, $a_1 = -\zeta \frac{s_{k-1}}{y_{k-1}^T s_{k-1}}$, $a_2 = \nabla f(x_k)$, $\zeta \left(\frac{C_k \|\nabla f(x_k)\|^2 s_{k-1} - (y_{k-1}^T s_{k-1})\nabla f(x_k)}{(y_{k-1}^T s_{k-1})^2} \right)$, and $a_4 = s_{k-1}$. Subsequently, we obtained determinant of Q_k as

$$\det(Q_k) = \left(\zeta \frac{\nabla f(x_k)^T s_{k-1}}{y_{k-1}^T s_{k-1}} - 1 \right)^2 + \zeta \frac{C_k \|\nabla f(x_k)\|^2 \|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2} - \zeta^2 \frac{\|\nabla f(x_k)\|^2 \|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2}. \tag{9.272}$$

However, the trace of a symmetric square matrix equals the sum of its eigenvalues; thus, we have

$$\mathrm{tr}(Q_k) = n - 2\zeta \frac{\nabla f(x_k)^T s_{k-1}}{y_{k-1}^T s_{k-1}} + \zeta \frac{C_k \|\nabla f(x_k)\|^2 \|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2} = \underbrace{1 + 1 + \cdots + 1}_{(n-2) times} + \eta_k^+ + \eta_k^-. \tag{9.273}$$

Accordingly,

$$\eta_k^+ + \eta_k^- = 2 - 2\zeta \frac{\nabla f(x_k)^T s_{k-1}}{y_{k-1}^T s_{k-1}} + \zeta \frac{C_k \|\nabla f(x_k)\|^2 \|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2}. \tag{9.274}$$

Therefore, from (9.266), (9.272), and (9.274), η_k^+ and η_k^- can be expressed as solutions of the following quadratic polynomial:

$$\eta_k^2 - (2 - 2\zeta p_k + \zeta C_k q_k)\eta_k + ((\zeta p_k - 1)^2 + \zeta(C_k - \zeta) q_k) = 0. \tag{9.275}$$

By applying the quadratic formula, we obtain

$$\eta_k^{\pm} = \frac{1}{2}\left[(2-2\zeta p_k+\zeta C_k q_k) \pm \sqrt{(\zeta C_k q_k - 2\zeta p_k)^2 + 4\zeta^2(q_k - p_k^2)}\right]. \quad (9.276)$$

This establishes the proof of (9.264) and (9.265). ■

Next, we must demonstrate that $\eta_k^+ > 0$ and $\eta_k^- > 0$. According to the Cauchy-Schwarz inequality, $q_k > p_k^2$, this proves that the discriminant is greater than zero. However, for $\eta_k^+ > 0$, we require

$$2 - 2\zeta \frac{\nabla f(x_k)^T s_{k-1}}{y_{k-1}^T s_{k-1}} + \zeta \frac{C_k \|\nabla f(x_k)\|^2 \|s_{k-1}\|^2}{(y_{k-1}^T s_{k-1})^2} > 0, \quad (9.277)$$

which is fulfilled by the calculation given below

$$C_k > 2\frac{(\nabla f(x_k)^T s_{k-1})(y_{k-1}^T s_{k-1})}{\|\nabla f(x_k)\|^2 \|s_{k-1}\|^2} - 2\frac{(y_{k-1}^T s_{k-1})^2}{\zeta \|\nabla f(x_k)\|^2 \|s_{k-1}\|^2}. \quad (9.278)$$

Besides, for $\eta_k^- > 0$, we require the following

$$\eta_k^- = \frac{1}{2}\left[(2-2\zeta p_k+\zeta C_k q_k) - \sqrt{(\zeta C_k q_k - 2\zeta p_k)^2 + 4\zeta^2(q_k - p_k^2)}\right] > 0. \quad (9.279)$$

Similarly, using some algebraic computations, we find that, $\eta_k^- > 0$ if

$$C_k > \zeta - \left(\frac{\sqrt{\zeta}(\nabla f(x_k)^T s_{k-1})}{\|\nabla f(x_k)\| \|s_{k-1}\|} - \frac{y_{k-1}^T s_{k-1}}{\sqrt{\zeta}\|\nabla f(x_k)\| \|s_{k-1}\|}\right)^2. \quad (9.280)$$

To ensure positive definiteness of the symmetric matrix Q_k, we present the following two parameters choices for C_k

$$C_k = \zeta - \psi\left(\frac{\sqrt{\zeta}(\nabla f(x_k)^T s_{k-1})}{\|\nabla f(x_k)\| \|s_{k-1}\|} - \frac{y_{k-1}^T s_{k-1}}{\sqrt{\zeta}\|\nabla f(x_k)\| \|s_{k-1}\|}\right)^2, \quad (9.281)$$

where $\zeta > 0$ and $\psi \leq 0$. Also, we obtain that $0 < \eta_k^- < \eta_k^+$. Hence all the eigenvalues of Q_k are positive real numbers. Accordingly Q_k is a positive definite matrix. Now, by defining $\eta_k^- = \eta \in (0,1)$ and multiplying (9.263), by $\nabla f(x_k)^T$,

we have

$$\nabla f(x_k)^T d_k = -\nabla f(x_k)^T Q_k \nabla f(x_k) \leq -\eta \|\nabla f(x_k)\|^2 < 0. \tag{9.282}$$

Thus, the sufficient descent condition is satisfied with $C_k > 0$, for all k.

On the other hand, if $F : \mathbb{R}^n \to \mathbb{R}^n$is the gradient mapping of some function $f : \mathbb{R}^n \to \mathbb{R}$ (i.e., $F = \nabla f$), then the problem (1.3) is simply the first-order necessary condition for the unconstrained optimization problem (1.4). Accordingly, to solve problem (1.3), we rewrite our proposed CG parameter and search direction as follows.

$$\beta_k^{UDDY} = \zeta \frac{\|F_k\|^2}{d_{k-1}^T y_{k-1}} - \zeta \frac{C_k \|F_k\|^2}{(d_{k-1}^T y_{k-1})^2} d_{k-1}^T F_k, \tag{9.283}$$

$$d_k = -F_k + \zeta \frac{\|F_k\|^2}{d_{k-1}^T y_{k-1}} s_{k-1} - \zeta \frac{C_k \|F_k\|^2 d_{k-1}^T F_k}{(d_{k-1}^T y_{k-1})^2} s_{k-1}, \tag{9.284}$$

and

$$C_k = \zeta - \psi \left(\frac{\sqrt{\zeta}(F_k^T s_{k-1})}{\|F_k\| \|s_{k-1}\|} - \frac{y_{k-1}^T s_{k-1}}{\sqrt{\zeta} \|F_k\| \|s_{k-1}\|} \right)^2. \tag{9.285}$$

where $\zeta > 0$ and $\psi \leq 0$. Furthermore, as a result of (9.282), we have

$$F_k^T d_k \leq -\eta \|F_k\|^2. \tag{9.286}$$

To improve a better direction towards the solution, we employed the approach of Waziri et al. [116] and suggested the following extended DY search direction.

$$d_k = -F_k + \beta_k^{EDDY} d_{k-1}, \tag{9.287}$$

with

$$\beta_k^{EDDY} = \zeta \frac{\|F_k\|^2}{\Psi_k} - \zeta \frac{C_k^* \|F_k\|^2}{\Psi_k^2} d_{k-1}^T F_k, \tag{9.288}$$

$$\Psi_k = \max\{\|F_k\| \|d_{k-1}\|, \zeta (d_{k-1}^T y_{k-1})\},$$

and

$$C_k^* = \zeta - \psi \left(\frac{\sqrt{\zeta}(F_k^T s_{k-1})}{v_k} - \frac{y_{k-1}^T s_{k-1}}{w_k} \right)^2. \tag{9.289}$$

with

$$v_k = \max\{\|F_k\|\|s_{k-1}\|, \sqrt{\zeta}(y_{k-1}^T s_{k-1})\},$$

$$w_k = \max\{\sqrt{\zeta}\|F_k\|\|s_{k-1}\|, y_{k-1}^T s_{k-1}\},$$

$\zeta > 0$, and $\psi \leq 0$.

The proposed method's algorithm is as follows:

Algorithm 8: Extended Descent DaiYuan Method (EDDY)

Input: Given $x_0 \in \Omega$, $d_0 = -F_0$, $\rho \in (0,1)$, $\sigma > 0$, $\varepsilon = 10^{-7}$, and $\xi > 0$, set $k = 0$.

Step 1: Compute F_k. If $\|F_k\| \leq \varepsilon$ then stop, else go to Step 2.

Step 2: Let $\alpha_k = \xi \rho^{m_k}$, with m_k being the smallest nonnegative integer m such that

$$-F(x_k + \alpha_k d_k)^T d_k \geq \sigma \alpha_k \|F(x_k + \alpha_k d_k)\|\|d_k\|^2. \tag{9.290}$$

Step 3: Set $z_k = x_k + \alpha_k d_k$.

Step 4: If $z_k \in \Omega$ and $\|F(z_k)\| \leq \varepsilon$, stop, otherwise go to Step 5.

Step 5: Compute the next iterate by

$$x_{k+1} = P_\Omega[x_k - \lambda_k F(z_k)], \tag{9.291}$$

where, $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step 6: Compute the search direction d_{k+1} using (9.287)-(9.289).

Step 7: Consider $k = k+1$ and go to Step 2.

9.3 Convergence Analysis of Algorithm 8 (EDDY)

In this subsection, the global convergence of Algorithm 2 is established.

Lemma 9.3.1 *Suppose assumptions (A1)-(A3) hold and let $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm 8, then $\{x_k\}$ and $\{z_k\}$ are bounded. In addition, we have*

$$\|d_k\| \leq M. \tag{9.292}$$

$$\lim_{k\to\infty} \|x_k - z_k\| = 0. \tag{9.293}$$

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{9.294}$$

Proof The proofs of (9.293) and (9.294) follow from Lemma 3.3.3. Consequently, there are some constants $m_1 > 0$ and $\kappa > 0$ such that

$$\|F(x_k)\| \leq m_1, \tag{9.295}$$

$$\|F(z_k)\| \leq \kappa, \tag{9.296}$$

and

$$\lim_{k\to\infty} \alpha_k \|d_k\| = 0. \tag{9.297}$$

Next, from (9.289)

$$\begin{aligned}
|C_k^*| &= \left| \zeta - \psi \left(\frac{\sqrt{\zeta}(F_k^T s_{k-1})}{v_k} - \frac{y_{k-1}^T s_{k-1}}{w_k} \right)^2 \right| \\
&\leq |\zeta| + |\psi| \left| \left(\frac{\sqrt{\zeta}(F_k^T s_{k-1})}{v_k} + \frac{y_{k-1}^T s_{k-1}}{w_k} \right)^2 \right| \\
&\leq \zeta + |\psi| \left| \left(\frac{\sqrt{\zeta}\|F_k\|\|s_{k-1}\|)}{\|F_k\|\|s_{k-1}\|} + \frac{y_{k-1}^T s_{k-1}}{y_{k-1}^T s_{k-1}} \right)^2 \right| \\
&= \zeta + |\psi|(\sqrt{\zeta}+1)^2.
\end{aligned} \tag{9.298}$$

Letting $Q = \zeta + |\psi|(\sqrt{\zeta}+1)^2$, we have

$$|C_k^*| \leq Q. \tag{9.299}$$

From (9.287), (9.288),(9.289), (9.295), and (9.299), we have

$$\begin{aligned}
\|d_k\| &= \| - F_k + \beta_k^{EDDY} d_{k-1}\| \\
&= \left\| -F_k + \left(\zeta \frac{\|F_k\|^2}{\Psi_k} - \zeta \frac{C_k^* \|F_k\|^2}{\Psi_k^2} d_{k-1}^T F_k \right) d_{k-1} \right\| \\
&\leq \|F_k\| + \left\| \left(\zeta \frac{\|F_k\|^2}{\Psi_k} - \zeta \frac{C_k^* \|F_k\|^2 d_{k-1}^T F_k}{\Psi_k^2} \right) d_{k-1} \right\| \\
&\leq \|F_k\| + \left\| \zeta \frac{\|F_k\|^2}{\Psi_k} d_{k-1} \right\| + \left\| \zeta \frac{C_k^* \|F_k\|^2 d_{k-1}^T F_k}{\Psi_k^2} d_{k-1} \right\| \\
&\leq \|F_k\| + \zeta \frac{\|F_k\|^2 \|d_{k-1}\|}{\Psi_k} + \zeta \frac{|C_k^*| \|F_k\|^3 \|d_{k-1}\|^2}{\Psi_k^2} \\
&\leq \|F_k\| + \zeta \frac{\|F_k\|^2 \|d_{k-1}\|}{\|F_k\| \|d_{k-1}\|} + \zeta \frac{Q \|F_k\|^3 \|d_{k-1}\|^2}{\|F_k\|^2 \|d_{k-1}\|^2} \\
&\leq \|F_k\| + \zeta \|F_k\| + \zeta Q \|F_k\| \\
&\leq (1 + \zeta + \zeta Q) \|F_k\| \\
&\leq (1 + \zeta + \zeta Q) m_1. \qquad (9.300)
\end{aligned}$$

We applied the Cauchy-Schwarz in the third inequality. By letting $M = (1 + \zeta + \zeta Q) m_1$, we have (9.292). ■

Theorem 9.3.2 *Suppose assumption (A1)-(A3) hold and* $\{x_k\}$ *be generated by Algorithm 8. Then*

$$\liminf_{k \to \infty} \|F_k\| = 0. \qquad (9.301)$$

Proof For the sake of contradiction, assume that (9.301) is not valid, then there exists $\delta_0 > 0$ such that

$$\|F_k\| \geq \delta_0 \quad \text{holds}, \quad \forall k > 0. \qquad (9.302)$$

Suppose that $\|d_k\| \neq 0$ unless at the solution, then there exist a constant, say δ_1

$$\|d_k\| \geq \delta_1. \qquad (9.303)$$

If the step length $\alpha_k \neq \xi$, then α_k, $\rho^{-1}\alpha_k$ does not satisfy (9.290) i.e.,

$$-F(x_k+\rho^{-1}\alpha_k d_k)^T d_k < \sigma\rho^{-1}\alpha_k\|d_k\|^2.$$

Thus, combining with (9.286) yields

$$\begin{aligned}\eta\|F_k\|^2 &\leq -F_k^T d_k \\ &= (F(x_k+\rho^{-1}\alpha_k d_k)-F(x_k))^T d_k - F(x_k+\rho^{-1}\alpha_k d_k)^T d_k \\ &\leq L\rho^{-1}\alpha_k\|d_k\|^2+\sigma\rho^{-1}\alpha_k\|d_k\|^2 \\ &= \alpha_k\|d_k\|(L+\sigma)\rho^{-1}\|d_k\|. \end{aligned} \tag{9.304}$$

Therefore, from (9.304), we have

$$\alpha_k\|d_k\| > \frac{\rho}{(L+\sigma)}\frac{\eta\|F_k\|^2}{\|d_k\|} \geq \frac{\rho}{(L+\sigma)}\frac{\eta\delta_0^2}{M}. \tag{9.305}$$

The inequality (9.305) contradicts (9.297). Therefore (9.301) holds. Thus, the proof is completed. ■

9.4 Applications in compressive sensing

This section is committed to applying the proposed algorithm (EDDY) to signal and image recovery problems in compressive sensing by solving the nonlinear monotone system of equations (2.45) which is the reformulation of the ℓ_1-norm regularization problems in compressive sensing (2.39).

To show the implementation of the EDDY scheme in compressive sensing, we compare it with the existing CHCG method [65]. We specified the parameters for the EDDY method as $\xi = 1$, $\sigma = 10^{-4}$, $\zeta = 10^{-4}$, $\psi = -0.5$, and $\rho = 0.9$. The parameters of the CHCG algorithm are taken as in [65]. The merit function $f(x) = \frac{1}{2}\|\vartheta - Mx\|_2^2 + \pi\|x\|_1$, in the experiment. We use the same continuation technique on the π parameter and only observe each method's convergence manners to solve with similar accuracy. In addition, the measuring signal starts the iterative process for the experiment, i.e., $x_0 = M^T\vartheta$, and ends when the objective function's relative

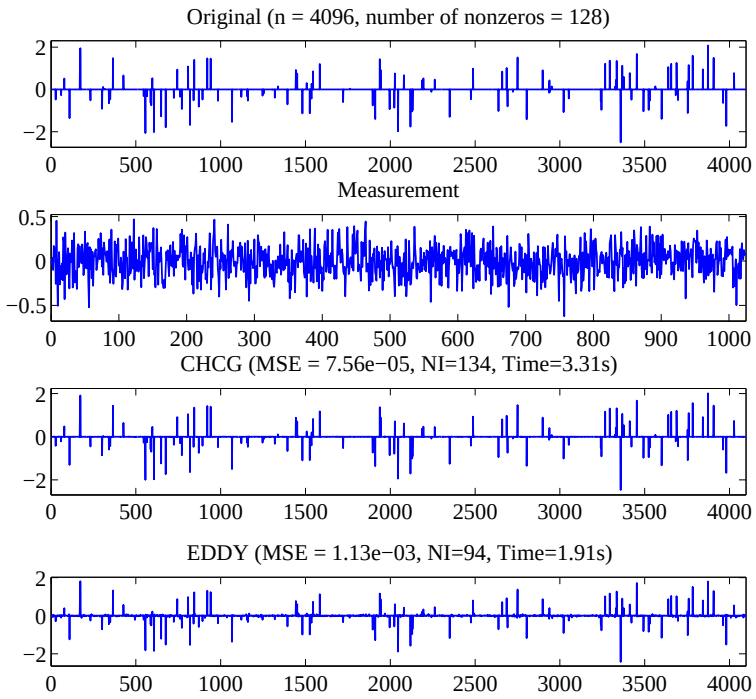

Figure 9.26: The original signal, the measurement, and the recovery signals using the EDDY and CHCG methods are shown from top to bottom.

change satisfies

$$\frac{|f(x_k) - f(x_{k-1})|}{|f(x_{k-1})|} < 10^{-5}, \tag{9.306}$$

where $f(x_k)$ is the function value at x_k.

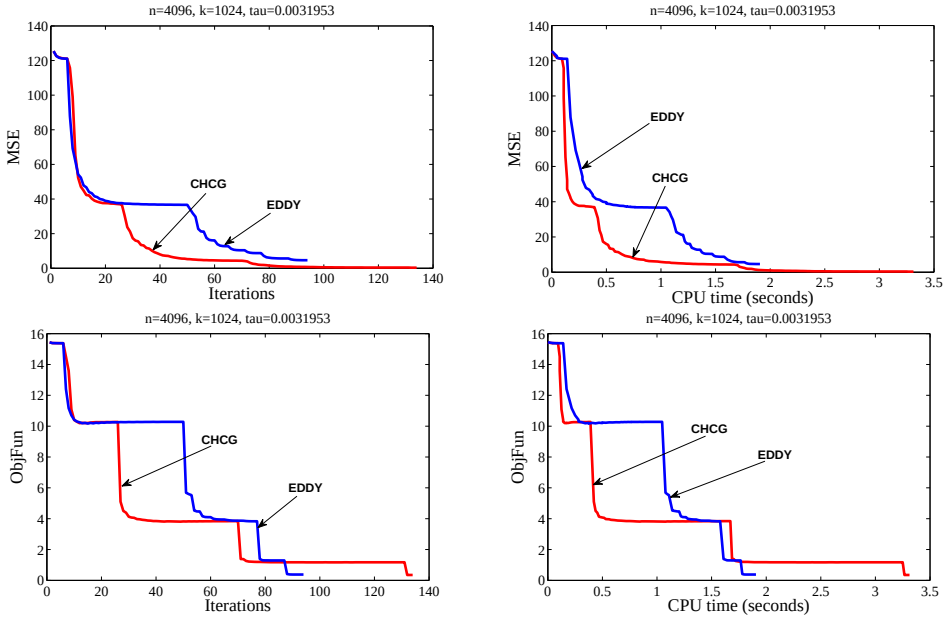

Figure 9.27: Comparison of the results of the EDDY and CHCG methods.

From Figure 9.26, almost precisely the disturbing signal has been restored by the EDDY and CHCG methods. However, the proposed method restored the disturbing signal with the least number of iterations and CPU time than the CHCG method. To visibly show these methods' interpretation, we plot four graphs to display the convergence behavior of the two approaches through their mean square error (MSE) and function evaluation results and the number of iterations and CPU time, respectively. As can be noticed from Figure 9.27, our proposed method shows much faster descent rates of MSE and function evaluations than the CHCG method. It can also be seen from the figures that the EDDY method requires fewer iterations and CPU time to recover the original signal compared to that needed for the CHCG method for the same procedure.

In the second experiment, we demonstrate the performance of our proposed algorithm for image de-blurring problems in greater detail. We specified the parameters for the EDDY method as taken in the signal processing experiment. However, the MDDYM methods' parameters are chosen the way they are selected from Wazir et al. [116]. As is customary, we estimated the restoration quality using the signal-to-noise ratio (SNR), defined as

$$\text{SNR} = 20 \times \log_{10}\left(\frac{\|\hat{x}\|}{\|\tilde{x}-\hat{x}\|}\right), \tag{9.307}$$

where $\hat{x}$ and $\tilde{x}$ are the original image and the restored image respectively. Furthermore, Structural Similarity (SSIM) index is used in this paper in order to measure the quality of the restored images [106]. The MATLAB implementation of the SSIM index can be obtained at `http://www.cns.nyu.edu/~lcv/ssim/`.

Figure 9.28: The original image (first row first column), the blurred image (first row second column), the restored image by EDDY (second row first column) and MDDYM (second row second column

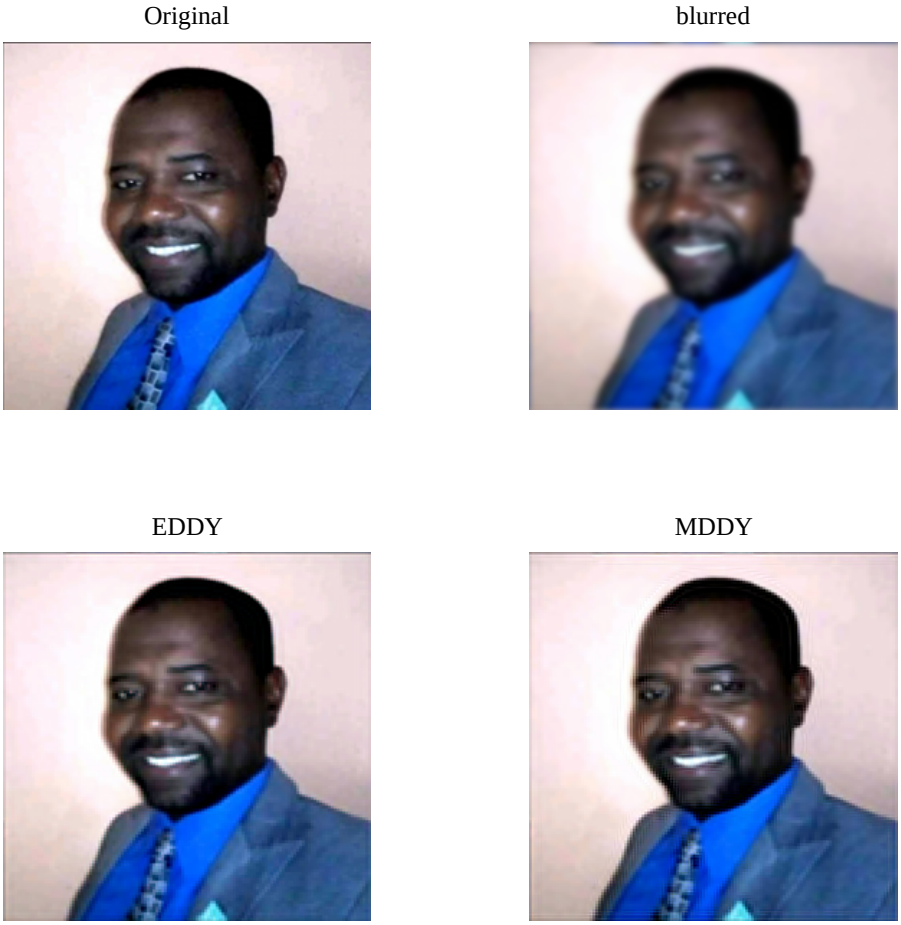

Figure 9.29: The original image (first row first column), the blurred image (first row second column), the restored image by EDDY (second row first column) and MDDYM (second row second column

Table 9.42: Numerical results for EDDY and MDDYM in image restoration

		EDDY				MDDYM			
IMAGE	SIZE	NI	TIME(s)	SNR	SSIM	NI	TIME (s)	SNR	SSIM
Majumder	256×256	23	2.48	28.07	0.91	40	4.27	26.46	0.89
Waziri	256×256	19	1.97	28.37	0.95	28	3.08	26.18	0.91

Figures 9.28 and 9.29 are generated to show the restoration results of two images obtained by the EDDY and MDDYM methods. Both methods are successful in restoring all photos, but the data in Table 9.42 indeed indicate that the proposed method has a higher efficacy. Because it has fixed the blurred images with fewer iterations and processor time, furthermore, it has shown from Table 9.42, the proposed method has the higher SNR and SSIM values. These indicated that the EDDYM method is more efficient at restoring blurred images.

9.5 Conclusion

In this chapter, an effective DY-type method was presented for system of monotone nonlinear equations and its applications signal and image processing in compressed sensing. Search direction of the scheme, which was combined with the projection method, was developed by analyzing eigenvalues associated with the iteration matrix of a modified DY-type version of the scheme due to Yuan and Guan (2005). Moreover, unlike the scheme by Yu and Guan, which converge for the parameter C in the interval $(\frac{1}{4}, \infty)$, the new method assures convergence for $C \in (0, \infty)$. The impact of this improvement is evident in the numerical performance exhibited by the new scheme when compared with some DY-type methods as well as in its application in compressed sensing problems. Global convergence has also been proven for the scheme.

CHAPTER NINE

Summary and Conclusion

9.1 Introduction

This chapter gives the summary and conclusion of the entire work, together with suggestions and recommendations for further research.

9.2 Summary

The derivative-free methods are type of iterative methods that are useful for unconstrained optimization problems. Because of their low memory requirements and global convergence properties, they are excellent choice for engineers and mathematicians dealing on large-scale systems of nonlinear equations.

The system of nonlinear equations and an unconstrained optimization problem are introduced in the first chapter of this thesis. The Newton Method, Quasi-Newton methods, and conjugate gradient methods are also discussed. It also includes objectives, statement of the problem, motivation of the study, scope and limitations, as well as definitions of some basic terms.

Chapter two included a review of the literature. Details of the proposed algorithms' derivations and convergence analyses, as well as numerical experiments, were given in Chapters three through eight.

Furthermore, the proposed methods presented in Chapters three, five, six, and eight are successfully applied to deal with the ℓ_1-norm regularization problem in compressive sensing, which outperform the existing methods in the literature.

9.3 Conclusion

This thesis presents derivative-free methods for solving system of nonlinear equations. Signal Recovery with Convex Constrained Nonlinear Monotone Equations through Conjugate Gradient Hybrid Approach (CHCG) method is presented in Chapter three and its performance is compared with PCG [73] and ETTCG [52] methods. In Chapter four, In Chapter four, accelerated dai-liao method for solving system of nonlinear equations (ADLP)is discussed and compared with PCG [73] and DLPM [1] methods. Furthermore, by comparing it to the IDFDD [59] and ADLCG [113] methods, the proposed method has been expanded to solve general system of nonlinear equations. In Chapter five, image deblurring problems through accelerated hager-zhang projection method for convex constrained monotone nonlinear equations (AHZP) is presented, and compared with the DLPM [1], MHZ2 [93], and SGCS [119] methods. In Chapter six, two double direction algorithms (DDDM and HDDM) for solving restricted monotone nonlinear equations are introduced. After that, the two proposed methods are numerically compared to the DDPM method [79]. Three-term Hager-Zhang projection method for monotone nonlinear equations is discussed in Chapter seven. The proposed method improved the numerical performance of the two Hager-Zhang methods proposed by Waziri et al. [112]. In chapter eight, an effective DY-type method was presented for system of monotone nonlinear equations as well as signal and image processing in compressed sensing. Search direction of the scheme, which was combined with the projection method, was developed by analyzing eigenvalues associated with the search direction matrix of a modified DY-type version of the Yuan and Guan (2005). Moreover, unlike the scheme by Yu and Guan, which converge for the parameter C in the interval $(\frac{1}{4},\infty)$, the new method assures convergence for $C \in (0,\infty)$.

The presented algorithms are shown to be globally convergent using basic assumptions. To demonstrate the performance and efficiency of the introduced methods, extensive numerical results were reported in tables and figures. The analysis concluded that the proposed methods are effective for dealing with large-scale problems.

Future Research

1. Many mathematicians use the inertial extrapolation approach in optimization theory to accelerate the convergence of iterative methods. Future research will also focus on improving the proposed methods for solving inertial-based derivative-free approaches for system of nonlinear equations.

2. Portfolio selection is an important aspect of financial management and investing decision making. Many types of portfolio selection challenges can be efficiently solved using optimization techniques (e.g. the classical Markowitz mean-variance model). In the future, we will propose optimization techniques that will tackle portfolio selection problems.

3. Solving the system of chemical equilibrium equations is among the most difficult numerical problems. However, monotone nonlinear equations can be used to address chemical equilibrium problems. As a result, we intend to use the proposed methods to address chemical equilibrium problems in the future.

LIST OF PUBLICATIONS

PUBLISHED PAPERS

S/N	Title	Journal	Index of journal	Remarks
1	Signal Recovery with convex constrained nonlinear monotone equations through conjugate gradient hybrid approach.	Mathematics and Computers in Simulation (ELSEVIER).	SCI Scopus. (Impact Factor: 2.463)	Published (2021).
2	On solving double direction method for convex constrained monotone nonlinear equations with Image restoration.	Computational and Applied Mathematics (SPRINGER).	Scopus SCIE (Impact Factor: 2.239)	Published (2021).
3	Motion Control of the Two Joint Planar Robotic Manipulators through Accelerated Dia-Liao Projection Method for Convex Constrained Monotone Nonlinear Equations.	Engineering Computations (EMERALD).	Scopus SCIE (Impact Factor: 1.59)	Published (2021).
4	On the Hybridization of the Step Length Method for Solving System of Nonlinear Equations.	Malaysian Journal of Mathematical Sciences.	Scopus	Published (2022).
5	Hybrid Accelerated Conjugate Gradient Method for Solving Nonlinear System of Equations.	Advances in Number Theory and Applied Analysis (SPRINGER).	**Book chapter**	Accepted (In press).

COMMUNICATED ARTICLES

S/N	Title	Journal	Index of journal	Remarks
1	Image Deblurring Problems through Accelerated Hager-Zhang Projection Method for Convex Constrained Monotone Nonlinear Equations.	FILOMAT.	Scimago (Q2) (Impact Factor: 0.95).	Under review. (Second round).
2	Three-term Hager-Zhang Projection Method for Monotone Nonlinear Equations.	Vietnam Journal of Mathematics (SPRINGER).	Scopus (Impact Factor: 0.55)	Under review.
3	On the Modified Dai-Yuan Method for Monotone Nonlinear Equations with Convex Constraint and its Applications.	Expert Systems with Applications (ELSEVIER).	SCIE Scopus (Impact Factor: 8.665)	Under Consideration.

REFERENCES

[1] Abubakar, A.B. and Kumam P., and Awwal A.M. (2018). A Descent Dai-Liao Projection Method for Convex Constrained Nonlinear Monotone Equations with Applications. *Thai Journal Mathematics*, 17(1): 128–152.

[2] Abubakar, A.B. and Kumam P. (2019). A descent Dai–Liao conjugate gradient method for nonlinear equations. *Numerical Algorithm*, 81: 197210.

[3] Abubakar, A.B., Kumam, P., and Awwal, A.M. (2019). Global convergence via descent modified three-term conjugate gradient projection algorithm with applications to signal recovery. *Results in Apply Mathematics*. 4: 1–20.

[4] Abdullahi, H., Halilu, A.S., and Waziri, M. Y. (2018). A Modified Conjugate Gradient Method via a Double Direction Approach for solving large-scale Symmetric Nonlinear Systems. *Journal of Numerical Mathematics and Stochastics*, 10(1): 32–44.

[5] Abubakar, A.B. and Kumam P., Mohammad, H., and Awwal A.M. (2020). A Barzilai-Borwein gradient projection method for sparse signal and blurred image restoration. *Journal of the Franklin Institute*, 357; 7266-7285.

[6] Aji, S., Kumam P., Siricharoen, P., Abubakar, A.B., Yahaya, M.M. (2020). A modified conjugate descent projection method for monotone nonlinear equations and image restoration. *IEEE Access*, 8: 158656-158665.

[7] Aminifard, Z. and Babaie-Kafaki, S. (2018). An optimal parameter choice for the Dai-Liao family of conjugate gradient methods by avoiding a direction of the maximum magnification by the search direction matrix. *4OR*, 1–14.

[8] Andrei, N. (2011). Open problems in conjugate gradient algorithms for unconstrained optimization. *Bulletin of the Malaysian Mathematical Science Society*, 34(2): 319–330.

[9] Andrei, N. (2016). An adaptive conjugate gradient algorithm for large-scale unconstrained optimization. *Journal of Computational and Applied Mathematics*, 292(1): 83-91.

[10] Andrei, N. (2017). A Dai-Liao conjugate gradient algorithm with clustering of eigenvalues. *Numerical Algorithm*, 77: 1273-1282.

[11] Awwal, A.M, Kumam, P., and Abubakar, A.B. (2019). A modified conjugate gradient method for monotone nonlinear equations with convex constraints. *Applied Numerical Mathematics*, 145: 507520.

[12] Awwal, A.M, Wang, L, Kumam, P, and Mohammad, H. (2020). A two-step spectral gradient Projection method for system of nonlinear monotone equations and image deblurring problems. Symmetry, 12: 874. doi:10.3390/sym12060874.

[13] Awwal, A.M, Kumam, P, Mohammad, H., Watthayu, W., Abubakar, A.B. (2020). A Perry-type derivative-free algorithm for solving nonlinear system of equations and minimizing ℓ_1 regularized problem. *Optimization*, doi: 10.1080/02331934.2020.1808647.

[14] Babaie–Kafaki, S. and Ghanbari, R. (2014). The Dai–Liao nonlinear conjugate gradient method with optimal parameter choices. *European Journal of Operation Research*, 234(3): 625–630.

[15] Babaie-Kafaki, S. Ghanbari, R. (2014). A descent family of Dai–Liao conjugate gradient methods. *Optimization Methods and Software*, 29(3): 583-591.

[16] Babaie-Kafaki, S. Ghanbari, R. (2014). Two modified three-term conjugate gradient methods with sufficient descent property. *Optimization Letters*, 8(8): 2285–2297.

[17] Babaie-Kafaki. (2014). On the sufficient descent condition of the Hager-Zhang conjugate gradient methods. *4OR*, 12(3): 285–292.

[18] Babaie-Kafaki, S. Ghanbari, R. (2015). Two optimal Dai-Liao conjugate gradient methods *Optimization*, 64(11): 2277-2287.

[19] Banham, M.R. and Katsaggelos, A.K., (1997). Digital image restoration. *IEEE Signal Processing Magazine*, 2(14): 24417.

[20] Barzilai, J. and Borwein, J. M. (1988). Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1): 141-148.

[21] Broyden, C.G. (1965). A class of methods for solving nonlinear simultaneous equations. *Mathematics of Computation*, 19: 577–593.

[22] Chan, C. L., Katsaggelos, A. K., and Sahakian, A. V. (1993). Image sequence filtering in quantum-limited noise with applications to low-dose fluoroscopy. *IEEE Transactions on Medical Imaging*, 12 (3): 610621.

[23] Cheng. W. (2007). A two-term prp-based descent method. *Numerical Functional Analysis and Optimization*, 28(11): 1217-1230.

[24] Chen, M. and Du,S. (2018). The smoothing FR conjugate gradient method for solving a kind of nonsmooth optimization problem with ℓ_1-norm. *Mathematical Problems in Engineering*, 2018; doi.org/10.1155/2018/5817931.

[25] Cheng, W. (2009). A PRP type method for systems of monotone equations. *Mathematical and Computer Modelling*, 50(1): 15-20.

[26] Crowder, H.P. and Wolfe, P. (1969). Linear convergence of the conjugate gradient method. *IBM J. Res. Dev.*, 16: 431-433.

[27] Cruz, W.L., Martínez, J. and Raydan, M. (2006). Spectral residual method without gradient information for solving large-scale nonlinear systems of equations. *Mathematics of Computation*, 75(255): 1429-1448.

[28] Cruz, W.L. and Raydan, M. (2003). Nonmonotone spectral methods for large-scale nonlinear systems. *Optimization Methods and Software*, 18(5): 583–599.

[29] Cruz, W L. (2017). A Spectral algorithm for large-scale systems of nonlinear monotone equations. *Numerical Algorithm*. DOI 10.1007/s1107s-017-0299-8.

[30] Dai, Y.H. and Kou, C.X. (2013). A nonlinear conjugate gradient algorithm with an optimal property and an improved wolfe line search. *SIAM. Journal of Optimization*, 23(1): 296–320.

[31] Dai, Y.H. and Liao, L.Z. (2001). New conjugacy conditions and related nonlinear conjugate gradient methods. *Applied Mathematics and Optimization*, 43(1): 87–101.

[32] Dai, Y.H. and Ni, Q. (2003). Testing different conjugate gradient methods for large-scale unconstrained optimization. *Journal of Computational Mathematics*, 21: 311–320.

[33] Dai, Z. and Wen, F. (2012). Another improved WeiYaoLiu nonlinear conjugate gradient method with sufficient descent property, *Apply Mathematics and Computation*, 218: 7421–7430.

[34] Dai, Y.H. and Yuan, Y. (1998). A class of Globally Convergent Conjugate Gradient Methods. *Research report ICM-98-030, Institute of Computational Mathematics and Scientific/Engineering Computing*, Chinese Academy of Sciences.

[35] Dai, Y.H. and Yuan, Y.X. (1999). A nonlinear conjugate gradient method with a strong global convergence property. *SIAM. Journal of Optimization*, 10: 177–182.

[36] Dai, Y.H. and Yuan, Y. (2001). An efficient hybrid conjugate gradient method for unconstrained optimization, Annals of Operations Research, 103 (2001), 33–47.

[37] Dai, Y.H. and Yuan, Y. (2003). A class of globally convergent conjugate gradient methods, *Science in China Series A-Mathematics*, 46: 251–261.

[38] Dai Z. and Wen. F. (2012). Another improved Wei-Yao-Liu nonlinear conjugate gradient method with sufficient descent property. *Applied Mathematics and Computation*, 218(14):7421–7430.

[39] Dauda, M.K., Mamat, M., Waziri, M.Y., Ahmad, F., and Mohamad, F.S.(2016). Inexact cg-method via sr1 update for solving systems of nonlinear equations. *Far East Journal of Mathematical Sciences*, 100(11): 1787-1804.

[40] Decker, D.W. and C.T. (1985). Broyden's methods for a class of problems having singular Jacobian at root, SIAM *Journal of Numerical Analysis*, 17: 566–574.

[41] Delladji, S. and Belloufi M., and Sellami, B. (2021). Behavior of the combination of PRP and HZ methods for unconstrained optimization. *Numerical Algebra, Control and Optimization*, 11(3): 377-389.

[42] Dennis, J.E. and More, J.J. (1974). A characterization of superlinear convergence and its application to quasi-Newton methods, *Mathematics of Computation*, 28: 549–560.

[43] Dennis, J.E and R.B Schnabel. (1983). Numerical method for unconstrained optimization and non-linear equations. *practice Hall, Englewood cliffs, NJ, USA*.

[44] Dolan, E. and Mor*é*, J. (2002). Benchmarking opyimization software with performance profiles, *Journal of Mathematical programming*. 91: 201–213.

[45] Fatemi, M. (2016). A new efficient conjugate gradientmethod for unconstrained optimization. *Journal of Computational and Applied Mathematics*, 300(1): 207–216.

[46] Fatemi, M. (2016). An optimal parameter for Dai-Liao family of conjugate gradient methods. *Journal of Optimization Theory and Application*, 169(2): 587-605.

[47] Fatemi M, Babaie-Kafaki, S.: Two extensions of the Daia–Liao method with sufficient desent property based on a penalization scheme. *Bulletin of Computational and Applied Mathematics*, 2016; 4(1) : 7-19.

[48] Figueiredo, M., Nowak, R. and Wright, S.J. (2007). Gradient projection for sparse reconstruction, application to compressed sensing and other inverse problems. *IEEE J-STSP IEEE Press, Piscataway, NJ*, 586-597.

[49] Fletcher, R.(1987). Practical Methods of Optimization vol. 1: unconstrained Optimization. *John Wiley Sons, New York.*

[50] Fletcher, R. and Reeves, C.M. (1964). Function minimization by conjugate gradient methods. *Journal of Computation*, 7: 149-154.

[51] Gilbert, J. C. and Nocedal, J. (1992). Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal of Optimization*, 2: 2142.

[52] Gao, P. and He, C. (2018). An efficient three-term conjugate gradient method for nonlinear monotone equations with convex constraints. *Calcolo*. 53: 17 pages.

[53] Gonglin, Y. (2009). A conjugate gradient method for unconstrained optimization problems. *International Journal of Mathematical Sciences*, Article ID 329623, 1–14.

[54] Gongli, Y. and Xiwen L. (2008). A new backtracking inexact BFGS method for symmetric nonlinear equations. *Journal of computers and mathematics with applications*, 55: 116-129.

[55] Griewank, A. (1986). The convergence of Broyden-like methods with a suitable line search. *Journal of the Australian Mathematical Society Series B*, 28: 75-92.

[56] Hager, W.W. and Zhang, H. (2005). A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal of Optimization*, 16(1): 170–192.

[57] Hager, W.W., Zhang, H. (2006). A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization* 2(1): 35-58.

[58] Halilu A.S., and Waziri M.Y. (2017) A transformed double step length method for solving large-scale systems of nonlinear equations. *Journal of Numerical Mathematics and Stochastics*, 9: 20–32.

[59] Halilu, A.S. and Waziri, M.Y. (2018). An improved derivative-free method via double direction approach for solving systems of nonlinear equations, *Journal of the Ramanujan Mathematical Society* 33(1): 75-89.

[60] Halilu A.S., Dauda,M.K., Waziri, M.Y., and Mamat, M. (2019). A derivative-free decent method via acceleration parameter for solving systems of nonlinear equations. *Open Journal of Science and Technology*, 2(3): 1-4.

[61] Halilu, A.S. Waziri, M.Y., Yusuf, I. (2020). Efficient matrix-free direction method with line search for solving large-scale system of nonlinear equations. *Yugoslav Journal of Operations Research*, 30(4): 399412.

[62] Halilu, A.S. and Waziri, M.Y. (2020). Solving systems of nonlinear equations using improved double direction method, *Journal of the nigerian mathematical society*, 32(2): 287-301.

[63] Halilu, A.S., and Waziri, M.Y. (2020). Inexact Double Step Length Method for Solving Systems of Nonlinear Equations. *Statistics, Optimization and Information Computing*, 8: 165–174.

[64] Halilu, A.S., Majumder, A., Waziri, M.Y., Abdullahi,H. (2020). Double direction and step length method for solving system of nonlinear equations. *European Journal of Molecular and Clinical Medicine*, 7(7): 3899–3913.

[65] Halilu, A.S., Majumder, A,. Waziri, M.Y., Ahmed, K. (2021). Signal recovery with convex constrained nonlinear monotone equations through conjugate gradient hybrid approach. *Mathematics and Computers in Simulation*, 187: 520-539.

[66] Hestenes, M.R. and Stiefel, E. (1952). Method of conjugate gradient for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6): 409-436.

[67] Hu, Y.F. and Storey, C. (1991). Global convergence result for conjugate gradient methods. *Journal of Optimization Theory and Applications*, 71: 399-405.

[68] Kačurovskii, R. I. (1960). On monotone operators and convex functionals.*Uspekhi Matematicheskikh Nauk* 15(4): 213215.

[69] Li D. and Fukushima M. (1999). A global and superlinear convergent Gauss-Newton based BFGS method for symmetric nonlinear equation. *SIAM Jornal of Numerical Analysis*, 37: 152-172.

[70] Li, Q. and Li, D.H. (2011). A class of derivative-free methods for large-scale nonlinear monotone equations. *IMA Journal of Numerical Analysis*, 31(4): 1625-1635.

[71] Li, J.K. and Liu, S.J. (2014). New hybrid conjugate gradient method for unconstrained optimization. *Applied Mathematics and Computation*, 245: 36-43.

[72] Liu, J. and Li. S. (2015). Spectral DY-type projection method for nonlinear monotone systems of equations. *Journal of Computational Mathematics*, 33(4): 341-355.

[73] Liu, J. K. and Li, S. J. (2015). A projection method for convex constrained monotone nonlinear equations with applications. *Computers and Mathematics with Applications*, 70(10): 2442-2453.

[74] Liu, Y. and Storey, C. (1991). Efficient Generalized Conjugate Gradient Algorithms, Part 1: Theory. *Journal of Optimization Theory and Applications*, 69: 129–137.

[75] Mascarenhas, W. F. (2004). The BFGS algorithm with exact line searches fails for nonconvex functions. *Journal of mathematical Programming*, 99(1): 49-61.

[76] Meintjes, K. and Morgan, A.P. (1987). A methodology for solving chemical equilibrium systems. *Apply Mathematics and Computation*, 22: 333361.

[77] Minty, G. J. (1962). Monotone (nonlinear) operators in Hilbert space. *Duke Mathematical Journal* 29(3): 341346.

[78] Mohammad, H. (2017). Barzilai-borwein-like method for solving large-scale non-linear systems of equations. *Journal of the nigerian mathematical society*, 35: 71–83.

[79] Mohammad, H. (2020). A descent derivative-free algorithm for nonlinear monotone equations with convex constraints. *Rairo-operations Research*, 54: 489–505.

[80] Mor*é*, J. and Stefan, M. : Benchmarking derivative-free optimization algorithms, SIAM J. Optimization, 2009; 20 (1): 172–191.

[81] Natasa, K. and Zorna, L. (2001). Newton-Like method with modification of rigtht-hand vector. *Mathematics of Computation*, 71: 237-250.

[82] Pang, J.S. (1986). Inexact Newton methods for the nonlinear complementarity problem, *Math. Program*, 1: 54-71.

[83] Perry, A. (1978). A modified conjugate gradient algorithm. *Operations Research Technical Notes*, 26(6): 1073–1078.

[84] Petrović, M.J. (2015). An accelerated double step size model in unconstrained optimization. *Applied Mathematics and Computation*, 250: 309-319.

[85] Petrović, M.J. (2019). Hybridization Rule Applied on Accelerated Double Step Size Optimization Scheme. *Filomat*, 33(3): 655665.

[86] Petrović, M.J., Stanimirović, P.S. (2014). Accelerated double direction method for solving unconstrained optimization problems. *Mathematical Problems in Engineering*, 1-8.

[87] Petrović, M.J., Stanimirović, P.S., Kontrec, N., Mladenović, J. (2018). Hybrid Modification of accelerated double direction method. *Mathematical Problems in Engineering*, 2018: 1-8.

[88] Polyak, B.T. (1969). The conjugate gradient method in extreme problems. *Computational Mathematics and Mathematical Physics*, 9: 94-112.

[89] Powell, M.J.D. (1977). Restart Procedures of the Conjugate Gradient Method. *Mathematical Programming*, 2: 241-254.

[90] Powell, M.J.D. (1983). Nonconvex minimization calculations and the conjugate gradient method. *Numerical Analysis*, Lecture Notes in Mathematics, 1066: 122141.

[91] Powell, M.J.D. (2000). On the convergence of the DFP algorithm for unconstrained optimization when there are only two variables. *Mathematical Programming*, 87: 281-301.

[92] Rao, S.S. (2009). Engineering Optimization Theory and Practice (4th Edition). *New Jersey: John Wiley and Sons, Inc*.

[93] Sabi'u, J., Shah, A., Waziri, M.Y., Ahmed, K. (2020). Modified Hager-Zhang conjugate gradient methods via singular value analysis for solving monotone nonlinear equations with convex constraint. *International Journal of Computational Methods*, https://doi:10.1142/S0219876220500437.

[94] Sabi'u, J., Shah, A., Waziri, M.Y. (2020) Two optimal Hager-Zhang conjugate gradient methods for solving monotone nonlinear equations. *Applied Numerical Mathematics*, 153: 217-233.

[95] Safeer H.K. (2013). A Picard-Mann hybrid iterative process. *Fixed Point Theory and Applications*, 69: 1-10.

[96] Shi, Z. and Shen, J. (2006). Convergence of nonmonotone line search method. *Journal of Computational and Applied Mathematics*, 193: 397-412.

[97] Solodov, M.V., and Svaiter, B.F. (1999). A globally convergent inexact Newton method for systems of monotone equations. In Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, *Springer*, 355-369.

[98] Stanimirović, P.S., Milovanović, G.V., Petrović, M.J., Kontrec, N. A. (2015) transformation of accelerated double step size method for unconstrained optimization. *Mathematical Problems in Engineering*, 1-8.

[99] Sun, W. and Yuan, Y.X. (2006). Optimization theory and methods: nonlinear programming. *Springer, New York Watkins DS (2002) Fundamentals of matrix computations, Wiley, New York*.

[100] Sun, X. (2015). A new spectral prp conjugate gradient method with suffcient descent property. *International Journal of Advances in Applied Mathematics and Mechanics*, 2(3): 51–59.

[101] Sun, M., Tian, M.Y. and Wang, Y.J. (2019). Multi-step discrete-time Zhang neural networks with application to time-varying nonlinear optimization. *Discrete Dynamics in Nature and Socociety*, Article ID 4745759, 114.

[102] Sun, M., Liu, J. and Wang, Y. (2020). Two Improved Conjugate Gradient Methods with Application in Compressive Sensing and Motion Control. *Mathematical Problems in Engineering*, 2020: 1-11.

[103] Touati-Ahmed, D. and Storey, C. (1990). Efficient hybrid conjugate gradient techniques. *Journal of Optimization Theory and Applications*, 64: 379–397.

[104] Wang C, Wang Y, Xu C. (2007). A projection method for a system of nonlinear monotone equations with convex constraints. *Mathematical Methods of Operation Research*, 66(1): 33-46.

[105] Wang C.W, Wang Y.J. (2009). A superlinearly convergent projection method for constrained systems of nonlinear equations. *Journal of Global Optimization*, 44(2): 283-296.

[106] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Trans, Image Process*, 13: 600–612.

[107] Waziri, M.Y., Leong, W.J., Hassan M.A. and Monsi, M. (2010). Jacobian computation-free Newton method for systems of nonLinear equations. *Journal of numumerical Mathematics and stochastic*, 2: 54-63.

[108] Waziri M.Y., Leong W.J., Hassan M.A. (2011). Jacobian-Free Diagonal Newton's Method for Solving Nonlinear Systems with Singular Jacobian. *Malaysian Journal of Mathematical Sciences*, 5: 241-255.

[109] Waziri, M.Y., Leong, W.J., Hassan, M.A., and Monsi, M. (2011). A low memory solver for integral equations of Chandrasekhar type in the radiative transfer problems. *Mathematical Problems in Engineering*, 1-12.

[110] Waziri, M.Y. and Majid, Z.A. (2013). An enhanced matrix-free secant mehod via predictor-corrector modified line search strategies for solving systems of nonlinear equations. *Journal mathematics and mathematical sciences*, 1-6.

[111] Waziri, M.Y. and Sabiu J. (2015). A derivative-free conjugate gradient method and its global convergence for symmetric nonlinear equations. *Journal of mathematics mathematical sciences*, 1-8.

[112] Waziri, M.Y., Ahmed, K. and Sabiu, J. (2019). A family of Hager–Zhang conjugate gradient methods for system of monotone nonlinear equations. *Applied Mathematics and Computation*, 361: 645–660.

[113] Waziri, M.Y., Ahmed K. and Sabiu, J. (2020). A Dai–Liao conjugate gradient method via modified secant equation for system of nonlinear equations. *Arabian Journal of Mathematics*, 9: 443457.

[114] Waziri, M.Y., Ahmed K., Sabiu, J., and Halilu, A.S. (2020). Enhanced Dai–Liao conjugate gradient methods for systems of monotone nonlinear equations. *SeMA Journal*, 1-37.

[115] Waziri, M.Y., Muhammad, H.U., Halilu, A.S., and Ahmed, K. (2020). : Modified matrix-free methods for solving system of nonlinear equations. *Optimization*, doi:10.1080/02331934.2020.1778689.

[116] Waziri, M.Y., Ahmad, K., Halilu, A.S., and Awwal, A.M. (2020). Modified Dai-Yuan iterative scheme for nonlinear systems and its application. *Numerical Algebra Control and Optimization*, doi:10.3934/naco.2021044

[117] William, L.C. and Raydan, M. (2003). Nonmonotone spectral methods for large-scale nonlinear systems. *Optimization Methods and Software*, 18(5): 583-599.

[118] Wu, L. and Sun, Z. (2012). New nonsmooth equations-based algorithms for ℓ_1-norm minimization and applications. *Journal of Applied Mathematics*, doi:10.1155/2012/139609.

[119] Xiao, Y., Wang, Q. and Hu, Q. (2011). Non-smooth equations based method for ℓ_1-norm problems with applications to compressed sensing. *Nonlinear Anal Theory Methods Appl.*, 74(11): 3570-3577.

[120] Xiao,Y. and Zhu, H. (2013). A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing. *Journal of mathematical analysis and application*, 405: 310319.

[121] Yan, Q.R., Pen, X.Z. and Li, D.H. (2010). A globally convergent derivative-free method for solving large-scale nonlinear monotone equations. *Journal of computational and applied mathematics*, 234(3): 649-657.

[122] Yao, S., Lu, X. and Wei, Z. (2013). A conjugate gradient method with global convergence for largescale unconstrained optimization problems, *Journal of Applied mathematics*, Article ID 730454.

[123] Ya-Xiang, Y. (2009). Subspace methods for large scale nonlinear equations and nonlinear least squares. *Optimization and Engineering*, 10(2): 207–218.

[124] Yu, Z., Lin, J., Sun, J., Xiao, Y., Liu, L. and Z. Li.(2009). Spectral gradient projection method for monotone nonlinear equations with convex constraints. *Applied numerical mathematics*, 59(10): 2416-2423.

[125] Yuan G, Lu X. :A new backtracking inexact BFGS method for symmetric nonlinear equations. (2008). *Computers and Mathematics with Applications*, 55: 116–129.

[126] Yu, G.H. and Guan, L.T.: New descent nonlinear conjugate gradient methods for large-scale unconstrained optimization. (2005). *Technical Report, Department of Scientific Computation and Computer Applications, Sun Yat-Sen University, Guangzhou, P.R. China*.

[127] Yu, G., Guan, L., and Chen, W.: Spectral conjugate gradient methods with sufficient descent property for large-scale unconstrained optimization. (2008). *Optimization Methods and Software*, 23: 275–293.

[128] Zarantonello, E. H. (1960). Solving functional equations by contractive averaging. *Technical Report 160. Madison, Wisconsin: U. S. Army Mathematics Research Center*.

[129] Zengxin, W., Guoyin, L., and Liqun Q. (2006). New nonlinear conjugate gradient formulas for large-scale unconstrained optimization problems.*Applied Mathematics and Computation*, 179(2): 407–430.

[130] Zhang, L. and Zhou, W. (2006). Spectral gradient projection method for solving nonlinear monotone equations. *Journal of Computational and Applied Mathematics*, 196(2): 478-484.

[131] Zheng, Y. and Zheng, B. (2017). Two new Dai-Liao-type conjugate gradient methods for unconstrained optimization problems. *Journal of Optimization Theory and Applications*, 175: 502509.

[132] Zhou, W., Li D.H. and Zhang, L. (2006). A descent modifed polak-ribiere-polyak conjugate gradient method and its global convergence. *IMA Journal of Numerical Analysis*, 26(4): 629-640.

[133] Zhou W. and Shen, D. (2014). An inexact PRP conjugate gradientmethod for symmetric nonlinear equations, *Numerical Functional Analysis and Optimization*, 35: 370388.