UNIVERSITÄT ZU LÜBECK
INSTITUTE OF MATHEMATICS
AND IMAGE COMPUTING

Numerical Methods for Constrained Systems of Equations

Numerische Verfahren für Gleichungssysteme mit Nebenbedingungen

Masterarbeit

verfasst am
Institute of Mathematics and Image Computing

im Rahmen des Studiengangs
Mathematik in Medizin und Lebenswissenschaften
der Universität zu Lübeck

vorgelegt von
Johannes Voigts

ausgegeben und betreut von
Prof. Dr. Jan Lellmann

mit Unterstützung von
Dr. Florian Mannel

Lübeck, den 15. März 2024

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Johannes Voigts

Zusammenfassung

Gleichungssysteme mit Nebenbedingungen sind eine wichtige Klasse von Problemen, für deren Lösung verschiedene Familien von numerischen Verfahren existieren. Verschiedene Verfahren eignen sich für verschiedene Arten von Problemen. In dieser Arbeit analysieren wir zwei Klassen von Verfahren theoretisch und numerisch. Die erste Klasse basiert auf dem LP-Newton Verfahren, welches sich für nichtglatte Probleme mit nicht-isolierten Lösungen eignet. Die zweite Klasse basiert auf dem Verfahren der konjugierten Gradienten, welche sich zusätzlich für hochdimensionale Probleme eignen, allerdings voraussetzen, dass die Probleme monoton sind. Unsere Analyse ergänzt bestehende theoretische Resultate und identifiziert in einigen Fällen Schwachstellen in diesen.

Abstract

Constrained systems of equations are an important class of problems for whose solution various families of numerical methods exist. Different methods are suited for different kinds of problems. In this thesis, we analyse two classes of methods for these systems theoretically and numerically. The first class is based on the LP-Newton method which is suited for nonsmooth problems with non-isolated solutions. The second class are Conjugate Gradient methods that are additionally suited for large scale systems, but require the problem to be monotone. Our analysis contributes further theoretical results and, in some cases, identifies weaknesses in existing ones.

Contents

1

Introduction

In this thesis, we consider numerical methods for *constrained systems of equations*, that is

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

for a (possibly nonlinear) function $F\colon \mathbb{R}^n \to \mathbb{R}^m$ and a set $\Omega \subseteq \mathbb{R}^n$. The system is constrained in the sense that all solutions must lie in the given set Ω, the *constraint set*. Throughout this thesis, this set is assumed to be nonempty, closed and convex. Further assumptions on the problem (1.1) will be provided as needed. The task is finding a x satisfying (1.1). The set of all such possible solutions will be denoted by Z, that is

$$Z \coloneqq \{x \in \Omega \,|\, F(x) = 0\}\,.$$

Constrained systems of equations occur in many applications, e.g., economic equilibrium problems [22], chemical equilibrium systems [54] and the power flow equations [82]. Constraints are often important where solutions might otherwise attain non-sensible values, e.g., negative concentration in chemistry problems or negative mass in physical problems. They can also be used to encode a-priori knowledge of our solutions in the problem. Constrained systems of equations can also be used to solve ℓ_1-norm regularized optimization problems in compressed sensing [26].

The systems that arise in these application are also often *monotone*, allowing the use of specialized solvers for these type of problems. Generally, different classes of methods are suited for different assumptions on F and Ω and make different trade-offs. When we impose stronger assumptions on F, then we clearly have more properties that we can exploit to create more powerful solvers. The benefits we get are usually improved speed or less memory consumption. On the downside, such specialisations limit the class of problems to which these methods are applicable.

An assumption on F of particular interest in this thesis is the *error bound condition*

$$\operatorname{dist}[x, Z] \leq \ell \|F(x)\|, \quad \forall x \in B_\delta(x^*) \tag{1.2}$$

for a $x^* \in Z$ and $\ell, \delta > 0$, where $\operatorname{dist}[x, Z]$ denotes the distance between x and the set Z. It is a main tool in achieving a convergence rate in all methods discussed in this thesis. For instance in Section 3.1, it serves as a generalization of regularity assumptions such

as the invertibility of $F'(x^*)$. In particular, (1.2) allows non-isolated solutions, while the invertibility of $F'(x^*)$ does not (see Theorem 3.3 and the example thereafter). Also note that (1.2) is also sensible in cases of $n \neq m$, as opposed to the invertibility of $F'(x^*)$.

1.1 Literature Review

Many numerical methods have been proposed for solving (1.1). We structure the review of the relevant literature according to the different types of methods that are contained in this thesis, starting with Newton-type methods, continuing with Spectral Gradient methods, and finishing with Conjugate Gradient methods. While we do not further analyse Spectral Gradient methods in this thesis, they are included in our numerical experiments because they are related to both Newton-type methods and Conjugate Gradient methods.

1.1.1 Newton-type Methods

Newton's method [32, 37, 56, 58] is one of the most well-known methods for solving (1.1) in the unconstrained case $\Omega = \mathbb{R}^n$. Its iterates $\{x_k\}$ are defined through the solution of a linear system of equations

$$F'(x_k)(x_{k+1} - x_k) = -F(x_k). \tag{1.3}$$

It has a fast convergence rate and can also be used for optimization problems by setting $F = \nabla f$ for the objective function $f\colon \mathbb{R}^n \to \mathbb{R}$. It has also been generalized in many ways, two of which are particularly relevant for this thesis. First, Semismooth Newton methods [38, 65, 73] use a generalized derivate instead of F' in (1.3) if F is not differentiable at x_k. Second, it is not necessary to solve (1.3) or its semismooth analogue exactly. Instead, for practical computations, it often suffices to compute an approximate solution. This approach is called *inexact* (semismooth) Newton method, see e.g., [19, 32] and [37, Chapter 6] for the differentiable case, as well as [25] and [73, Section 3.2.4] for the semismooth case. Usually, the approximation quality required for (1.3) needs to satisfy

$$\|F(x_k) + F'(x_k)(x_{k+1} - x_k)\| \leq \eta_k \|F(x_k)\| \tag{1.4}$$

for a so-called *forcing sequence* $\{\eta_k\} \subseteq [0, 1)$.

These Newton-type methods have influenced the development of the *LP-Newton method* by Facchinei et al. [24], which is one of the main subjects of this thesis (see Section 3.1). The LP-Newton method is a novel approach that replaces the linear system of equations (1.3) by a Linear Program, which is designed in such a way that its solutions x_{k+1} satisfy $x_{k+1} \in \Omega$, thereby ensuring feasibility for all iterates. This Linear Program also contains a condition related to (1.4) and involves the linearization of F around the current iterate x_k. For this purpose, however, it does not assume that $F'(x_k)$ exists, but rather allows the usage of a generalized derivative; this is analogue to how semismooth Newton methods generalize Newton's method. This approach allows possibly nonsmooth F, while maintaining the quadratic convergence of the (semismooth) Newton Method. Furthermore, it also allows non-isolated solutions by employing the error bound condition.

As the LP-Newton method only converges locally, Fischer et al. [28] created a variant of LP-Newton that converges globally. They achieve this by adding a backtracking line search after each Linear Program. This line search involves a custom measure of the directional descent for $\|F(\cdot)\|$. While the algorithm converges globally, it still preserves the local quadratic convergence of LP-Newton.

A different globalization strategy has been employed by Becher et al. [7] who use a trust-region strategy to solve (1.1) for a piecewise smooth F and a polyhedral set Ω. They establish a Q-order of convergence (see Section 2.2 for the definition of Q-order) under *Hölder metric subregularity* of F and *Hölder continuity* of the derivative of the selection mapping.

Another important class of methods to tackle (1.1) in the unconstrained case are Quasi-Newton methods [20, 50]. In these methods, the derivative in (1.3) is replaced by an approximation B_k. This is advantageous in situations where the derivative is expensive to compute or unavailable. Usually, they are updated in each step iteratively. Often, it is possible to directly update B_k^{-1} itself, which greatly simplifies the calculation of the solution of (1.3), as it is not required to solve linear systems. Classic examples for this include the famous BFGS Method [56, Section 6.1] for optimization problems, and Broyden's Method [9] for systems of equations. With Quasi-Newton methods, it is possible to achieve superlinear convergence [11]. While the previously stated Quasi-Newton methods are designed for the unrestricted case $\Omega = \mathbb{R}^n$, mathematicians have also proposed variants that work in the restricted case to extend the aforementioned benefits to these problems. One approach by Marini et al. [49] uses the approximate norm descent condition [44]

$$\|F(x_{k+1})\| \leq (1 + \eta_k)\|F(x_k)\|$$

for a non-negative summable sequence of real numbers $\{\eta_k\}$, allowing $\{\|F(x_k)\|\}$ to not be monotonically decreasing. They employ a line-search strategy that in each step either enforces sufficient decrease of $\|F(\cdot)\|$ or approximate norm decrease. The condition $x \in \Omega$ is enforced by a projection onto Ω. The method can be used with various Quasi-Newton approximations of F', e.g., the Broyden-Schubert update [10, 68], the Bogle-Perkins Update [8], and the Inverse Column Update [51]. They also provide an upper bound on the number of needed iterations.

The LP-Newton method has also been extended by Martínez and Fernández [52] to employ Quasi-Newton approximations of the derivative to remove the need to calculate derivatives in each step while achieving linear convergence. The authors have later improved the method to obtain superlinear convergence [53]. To prove convergence, the authors make an assumption on the Quasi-Newton approximations that they themselves however acknowledge may not hold. We will discuss this assumption in Section 5.1.

1.1.2 Spectral Gradient Methods

Spectral Gradient algorithms can be seen as a variant of Quasi-Newton methods, where the Jacobian (or the Hessian in the optimization case) is approximated via $B_k = \lambda_k^{-1} I$, where I is the identity matrix. Thus, the search directions d_k can be cheaply calculated by $d_k = -\lambda_k F(x_k)$ (or $d_k = -\lambda_k \nabla f(x_k)$) and the next iterate is then given by $x_{k+1} =$

$x_k + \alpha_k d_k$ with a suitable α_k. As d_k is chosen as a multiple of the negative gradient in the optimization case, it can also be seen as a variant of the Gradient Descent method (see e.g., [12]) introduced by Cauchy [13]. The parameter λ_k^{-1} contains spectral information about the Jacobian or Hessian, hence the name Spectral Gradient methods.

The concept of Spectral Gradient algorithms has first been introduced by Barzilai and Borwein [6], who chose the parameter λ_k such that B_k approximates the *secant equation*

$$B_k(x_k - x_{k-1}) = \nabla f(x_k) - \nabla f(x_{k-1})$$

as closely as possible. They applied the algorithm to quadratic optimization problems. Inspired by this, La Cruz and Raydan [43] introduced the SANE method to solve nonlinear unconstrained systems of equations. This method however includes calculations of the Jacobian. To alleviate this problem, La Cruz et al. [41] proposed a variant of SANE called DF-SANE, which is indeed derivative-free. They further employ a non-monotone line search strategy based on the approximate norm descent condition by Li and Fukushima [44].

Building upon these two methods, La Cruz [40] later introduced the PSANE method, which extends SANE and DF-SANE to the constrained problem (1.1). They enforce the constraint $x \in \Omega$ by employing a projection onto Ω in a systematic way.

We are also aware of a modification of PSANE by Morini et al. [55] called PAND. They employ a line search strategy similar to the previously discussed Quasi-Newton method by Marini et al. [49]. This relation is not a coincidence, as Morini and Porcelli are authors of both papers.

1.1.3 Conjugate Gradient Methods

Another class of methods for the solution of (1.1) are the Conjugate Gradient (CG) methods. The original algorithm has been introduced by Hestenes and Stiefel [36] to find solutions of strictly convex quadratic optimization problems, i.e., problems of the form

$$\min_{x \in \mathbb{R}^n} x^\top A x - b^\top x \tag{1.5}$$

with a symmetric positive definite matrix $A \in \mathbb{R}^{n \times n}$ and a vector $b \in \mathbb{R}^n$. Solving (1.5) is equivalent to solving $Ax = b$. Two important properties of this algorithm are that it terminates in a finite number of steps and that all operations in each step are linear time except the evaluation of $x \mapsto Ax$ (for further details, see Chapter 4). The search directions are of the form

$$d_k = \begin{cases} -\nabla f_k & \text{if} \quad k = 0 \\ -\nabla f_k + \beta_k d_{k-1} & \text{if} \quad k > 0 \end{cases}$$

with $\nabla f_k \coloneqq \nabla f(x_k)$. Going forward, we will also use the common abbreviations $y_k \coloneqq \nabla f_{k+1} - \nabla f_k$ and $s_k \coloneqq x_{k+1} - x_k$. The *CG parameter* β_k is chosen such that $d_i^\top A d_j = 0$ for $i \neq j$, which means that different directions are orthogonal to each other wrt. the scalar product induced by A. The iterates are then updated by $x_{k+1} = x_k + \alpha_k d_k$ with a positive scalar α_k which is usually determined through a line search.

The CG method has first been extended by Fletcher and Reeves [30] to general convex optimization problems

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1.6}$$

with the goal to keep the original CG method's cheap iterations in terms of speed and memory. Others have followed, which led to an emergence of various CG methods employing different CG coefficients. A collection of classical choices can be found in Table 1.1, where $\beta_{k+1}^{\mathrm{HS}}$ also denotes the coefficient of the original CG method. All of these coefficients coincide

Coefficient	Source
$\beta_{k+1}^{\mathrm{HS}} = \dfrac{\nabla f_{k+1}^\top y_k}{y_k^\top d_k}$	Hestenes and Stiefel [36]
$\beta_{k+1}^{\mathrm{FR}} = \dfrac{\nabla f_{k+1}^\top \nabla f_{k+1}}{\nabla f_k^\top \nabla f_k}$	Fletcher and Reeves [30]
$\beta_{k+1}^{\mathrm{PRP}} = \dfrac{\nabla f_{k+1}^\top y_k}{\nabla f_k^\top \nabla f_k}$	Polak and Ribière [61] and Polyak [62]
$\beta_{k+1}^{\mathrm{CD}} = \dfrac{\nabla f_{k+1}^\top \nabla f_{k+1}}{-\nabla f_k^\top d_k}$	Fletcher [29] (Conjugate Descent)
$\beta_{k+1}^{\mathrm{LS}} = \dfrac{\nabla f_{k+1}^\top y_k}{-\nabla f_{k+1}^\top d_k}$	Liu and Storey [48]
$\beta_{k+1}^{\mathrm{DY}} = \dfrac{\nabla f_{k+1}^\top \nabla f_{k+1}}{y_k^\top d_k}$	Dai and Yuan [18]
$\beta_{k+1}^{\mathrm{DL}} = \dfrac{\nabla f_{k+1}^\top y_k}{y_k \top d_k} - t\dfrac{\nabla f_{k+1}^\top s_k}{y_k \top d_k}$	Dai and Liao [17]

Table 1.1: Various common choices for β_k. The last coefficient $\beta_{k+1}^{\mathrm{DL}}$ contains an parameter $t > 0$, those optimal choice remains subject of research [3].

in the quadratic case. Hence, they are sensible generalizations. More on CG methods for optimization problems can be found, for instance, in [4, 35, 56].

A notable property of CG methods is their connection to Quasi-Newton methods. Perry [60] expressed the Hestenes-Stiefel search direction as a matrix-vector product $d_k = -Q_k \nabla f_k$. Additionally, Shanno [69] showed that resetting the Quasi-Newton approximation in the BFGS method to the identity matrix in each step yields exactly the Conjugate Gradient method. This also led both to develop the *self-scaled memoryless BFGS method*, where at each step, the Quasi-Newton approximation is reset to a scalar multiple of the identity matrix [59, 70].

Closely related to this method is the very influential CG_DESCENT Conjugate Gradient method by Hager and Zhang [34]. Its coefficient is a modification of $\beta_{k+1}^{\mathrm{HS}}$ given by

$$\beta_{k+1}^{\mathrm{HZ}} = \frac{\nabla f_{k+1}^\top}{y_k^\top d_k}\left(y_k - 2\frac{\|y_k\|^2}{y_k^\top d_k} d_k\right),$$

which again coincides with the other coefficients in the quadratic case. The resulting search direction leads to guaranteed descent independent of the line search. The method reaches high performance and achieves this through a Wolfe line search [56, 80, 81] among other things.

In the case of f being convex and continuously differentiable, solving (1.6) is equivalent to solving $\nabla f(x) = 0$. As such, CG methods can also be viewed as methods for solving systems of equations. In fact, CG methods can be generalized to general systems $F(x) = 0$,

as long as F is continuous and *monotone* (see Definition 4.1 and Lemma 4.2). In that case, the coefficients of Table 1.1 can be repurposed by replacing ∇f_k with $F_k \coloneqq F(x_k)$.

An important technique that contributed to the success of CG methods for nonlinear equations has been a projection technique developed by Solodov and Svaiter [71] which they used to achieve global convergence for their inexact Newton method by exploiting properties of monoton functions. Instead of using $x_k + \alpha_k d_k$ as the next iterate directly, it would first be projected onto a hyperplane separating the iterate x_k from the solution set. This method has been further improved by Wang et al. [77] who developed a method for monoton equations based on [71]. This has then, again, be improved by Wang and Wang [76] to achieve superlinear convergence. However, their method was not matrix free and thus unsuitable for large scale systems.

This has then motivated the use of the hyperplane projection technique in CG methods to solve nonlinear equations. The first implementation has been by Cheng [14] in their method based on the β_k^{PRP} coefficient. They managed to prove global convergence under a Lipschitz assumption.

There has further been an approach by Li and Wang [46] to solve unconstrained nonlinear systems based on β_{k+1}^{FR} without using hyperplane projection. This method is also applicable for non-monotone F, however it mandates that F is continuously differentiable and that $F'(x)$ is symmetric for all $x \in \mathbb{R}^n$. The Jacobian however does not need to be calculated. Thus, the method remains efficient for large scale systems.

Finally, the idea of Cheng [14] to implement the hyperplane projection in CG methods has then been extended to the constrained problem (1.1) by Xiao and Zhu [83]. Their method is based on the powerful CG_DESCENT and also achieved global convergence under a Lipschitz assumption. The constraint $x \in \Omega$ is enforced by a projection onto Ω after the hyperplane projection. They tested their method with a signal and image reconstruction application in compressed sensing. Their method has further been improved by Liu and Li [47]. Note that many articles demonstrate improvements on CG methods for constrained systems in numerical experiments only, i.e., without providing rigorous theoretical justification of these improvements.

In 2013, Dai and Kou [16] proposed a new conjugate gradient method for minimization problems. Their search direction is chosen as a vector on a one-dimensional manifold that is closest to the direction of the self-scaled memoryless BFGS method by Perry [59] and Shanno [70]. Their new CG coefficient is given by

$$\beta_{k+1}^{\text{DK}} = \frac{\nabla f_{k+1}^\top y_k}{y_k \top d_k} - \left(\tau_k + \frac{\|y_k\|^2}{s_k^\top y_k} - \frac{s_k^\top y_k}{\|s_k\|^2}\right) \frac{\nabla f_{k+1}^\top s_k}{y_k \top d_k},$$

with a scalar parameter τ_k which corresponds to the self-scaling parameter in the self-scaled memoryless BFGS method. This new coefficient can be seen as a special case of the Dai-Liao coefficient β_{k+1}^{DL} with $t = \left(\tau_k + \frac{\|y_k\|^2}{s_k^\top y_k} - \frac{s_k^\top y_k}{\|s_k\|^2}\right)$. They combined their method with an improved Wolfe line search.

The Dai-Kou approach was first applied to constrained systems of equations by Ding et al. [21]. They used the hyperplane projection technique [71] and established linear convergence for their scheme under an error bound condition. Their choice for τ_k is taken as one of the effective choices in [16] and [57] or their convex combination. Numerically, their method is very competitive.

Recently, Waziri et al. [79] proposed a modified Dai-Kou scheme where they use a modification of the vector y_k by Li and Fukushima [45] in the formula for $\beta_{k+1}^{\mathrm{DK}}$. The method was tested with an application in signal and image reconstruction. They have further improved this approach in Ahmed et al. [2]. An important change is the addition of a scalar multiple of (the modified) y_k as a third term to the search direction d_{k+1}. A notable property of this algorithm is that the search directions are of the form $d_k = -Q_k F_k$ with a symmetric matrix Q_k. They have proven linear convergence for their scheme and it is performing very well numerically. We will analyse this method further in Sections 4.3.1 and 5.3.

2

Preliminaries

2.1 Basic Concepts and Notation

For this entire thesis, let $F\colon \mathbb{R}^n \to \mathbb{R}^m$ be the continuous function for which we want to solve (1.1). In practice, for most methods we have the case $m = n$. Notable exceptions are the methods proposed in Sections 3.1 and 3.2, i.e., the LP-Newton method and its variants.

The set Ω will always denote a closed convex subset of $\mathbb{R}^n$. It encompasses our constraints on the solutions in (1.1). By Ω being *convex*, we mean that

$$\forall x, y \in \Omega, \forall \tau \in [0,1]: \quad (1-\tau)x + \tau y \in \Omega$$

holds, that is for every two points in Ω, the line segment connecting the two lies completely within Ω. By Z, we will denote our set of solutions for (1.1), i.e.,

$$Z \coloneqq \{z \in \Omega \mid F(z) = 0\}.$$

For this entire thesis, this set is assumed to be nonempty. As F is continuous and Ω closed, the set $Z = F^{-1}(\{0\}) \cap \Omega$ is also closed. This is an important property as we will see in the following.

First, we will equip $\mathbb{R}^n$ and $\mathbb{R}^m$ each with a norm, that we will denote with $\|\cdot\|$ in both cases. Which of these two norms is used will be clear from context. Now, let $x \in \mathbb{R}^n$ and C be a subset of $\mathbb{R}^n$. Then, the *distance* $\operatorname{dist}[x, C]$ of the point x to the set C is given by

$$\operatorname{dist}[x, C] \coloneqq \inf_{y \in C} \|x - y\|.$$

When the set C is closed, the infimum is attained, that is for all $x \in \mathbb{R}^n$ there exists an $\bar{x} \in C$ such that $\operatorname{dist}[x, C] = \|x - \bar{x}\|$. This property is one of the reasons we need the closedness of Z. If we consider a set that is additionally convex, e.g., Ω, then $\bar{x}$ is additionally unique when we choose $\|\cdot\|$ as the euclidean norm [33, Section 2.1.3]. This allows the definition of the *projection map* $P_\Omega\colon \mathbb{R}^n \to \Omega$ onto Ω by defining $P_\Omega(x) = \bar{x}$ for $\operatorname{dist}[x, \Omega] = \|x - \bar{x}\|$ with $\bar{x} \in \Omega$.

Next, an important concept in smooth as well as nonsmooth analysis is Lipschitz continuity. The function F is called *Lipschitz continuous* or just *Lipschitz*, when there exists an $L > 0$, such that

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

While this is a global property, there is also a weaker local variant of Lipschitz continuity. The function F is called *locally Lipschitz continuous* or just *locally Lipschitz* at a point $z \in \mathbb{R}^n$ if there exist $L > 0$ and $\varepsilon > 0$ such that

$$\|F(x) - F(y)\| \leq L\|x - y\|, \quad \forall x, y \in B_\varepsilon(z).$$

The set $B_\varepsilon(z)$ denotes the closed ε-neighborhood around z. An example of a Lipschitz continuous function is the projection map P_Ω, which is Lipschitz with $L = 1$ [33, Section 2.1.3], that is

$$\|P_\Omega(x) - P_\Omega(y)\| \leq \|x - y\|, \quad \forall x, y \in \mathbb{R}^n.$$

Finally, the last concept we introduce in this section is the *operator norm* [39, page 26]. It allows us to construct a norm on the matrix space $\mathbb{R}^{m\times n}$ from the norms on $\mathbb{R}^n$ and $\mathbb{R}^m$. The *operator norm* of $A \in \mathbb{R}^{m\times n}$ is defined by

$$\|A\| \coloneqq \sup_{x\neq 0} \frac{\|Ax\|}{\|x\|}. \tag{2.1}$$

An important property of this norm is that it satisfies the inequality

$$\|Ax\| \leq \|A\|\,\|x\|$$

for all $x \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m\times n}$.

2.2 Rates of Convergence

An important notion when analysing the convergence properties of an algorithm is its *convergence rate*. Naturally, a fast convergence rate is desirable, as we then need fewer iterations to reach a solution. There are various classifications of different convergence rates. The definitions in this section follow [56, 58].

Let $\{x_k\} \subseteq \mathbb{R}^n$ be a sequence that converges to some $x^* \in \mathbb{R}^n$. Also, let $\|x_k - x^*\| \neq 0$ for sufficiently large k. The sequence $\{x_k\}$ is said to converge *Q-linearly*, when

$$r \coloneqq \limsup_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} < 1.$$

This means that the distance to the solution decreases by at least the constant factor $r \in (0, 1)$. A faster convergence rate occurs when the limit r is zero,

$$\limsup_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0.$$

In that case, we say that the sequence converges *Q-superlinearly*.

A more rapid convergence rate occurs when there exists a $p > 1$, such that

$$\limsup_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p} < \infty.$$

The number p is called the *Q-convergence order* of $\{x_k\}$. A higher convergence order is better, as a convergence order of p implies a convergence order of q for all $q \in (1, p)$. In the special case of $p = 2$, we also call the convergence rate *Q-quadratic*.

There is also a notion weaker than the "Q-" family of convergence rates, that is the "R-" family of convergence rates. The sequence $\{x_k\}$ is said to converge *R-linearly*, if there exists a sequence $\{\varepsilon_k\}$ of non-negative real numbers that converge Q-linearly to zero, such that for all k

$$\|x_k - x^*\| \leq \varepsilon_k.$$

Q-linear convergence rates are stronger that R-linear rates in the sense that they mandate sufficient decrease of the distance to x^* for *all* k large enough, while an R-linear convergence rate allows occasional deviation, as long as the distance admits sufficient decrease overall. We could also define "R-" versions of the other convergence rates, however we only encounter R-linear rates in this thesis.

Usually, we will drop the "Q-" prefix for improved readability and only include it for contrast with "R-" convergence rates.

2.3 Generalized Derivatives

In this thesis, we are particularly interested in those cases where F fails to be "smooth enough" or to be differentiable. For those cases it is often desirable to have weaker alternative notions of a derivative. In this thesis, we will mention the following.

Definition 2.1. (cf. [38, Sect. 1.3]) For $F\colon \mathbb{R}^n \to \mathbb{R}^m$ locally Lipschitz, the *B-subdifferential* (Bouligand-subdifferential) at a point $x \in \mathbb{R}^n$ is defined as

$$\partial_B F(x) \coloneqq \left\{ \lim_{k\to\infty} F'(x_k) \mid \lim_{k\to\infty} x_k = x,\ \{x_k\} \subseteq \mathcal{D}_F \right\},$$

where $\mathcal{D}_F \subseteq \mathbb{R}^n$ is the set of points where F is differentiable. Then, *Clarke's generalized Jacobian* [15, Definition 2.6.1] at z is defined as

$$\partial F(x) \coloneqq \operatorname{conv} \partial_B F(x).$$

Note that Rademacher's Theorem states that when F is locally Lipschitz, it is differentiable almost everywhere (see [66, Theorem 9.60]). That is, $\mathbb{R}^n \setminus \mathcal{D}_F$ has Lebesgue-measure zero, and thus for all $x \in \mathbb{R}^n$ a sequence $\{x_k\} \subseteq \mathcal{D}_F$ exists such that

$$\lim_{k\to\infty} x_k = x.$$

Definition 2.2. The function F is called *directionally differentiable* at $x \in \mathbb{R}^n$ if

$$\lim_{t\to 0^+} \frac{F(x + th) - F(x)}{t} =: F'(x; h)$$

exists for all $h \in \mathbb{R}^n$.

Definition 2.3. $F\colon \mathbb{R}^n \to \mathbb{R}^m$ is called *semismooth* [65] at x if it is locally Lipschitz in x and

$$\lim_{\substack{V \in \partial F(x+th') \\ h' \to h, t \to 0^+}} Vh' \tag{2.2}$$

exists for all $h \in \mathbb{R}^n$. F is called *strongly semismooth* [64, Definition 2.3] at x if additionally

$$Vh - F'(x;h) = O(\|h\|^2) \tag{2.3}$$

holds for all $V \in \partial F(x+h)$ and $h \to 0$.

Note that the directional derivative $F'(x;h)$ exists in (2.3) when F is semismooth at x and coincides with (2.2), see [65, Proposition 2.1].

3

Newton-type Methods

Newton's Method is one of the most classic methods for solving

$$F(z) = 0$$

and is well known for its fast convergence rate. In each step, we acquire the step by solving a linear system of equations involving the derivative of F.

To specify Newton's method, let $F\colon \mathbb{R}^n \to \mathbb{R}^n$ be differentiable and $z^0 \in \mathbb{R}^n$. The sequence of iterates $\{z^k\}$ is recursively defined by

$$\begin{aligned} F'(z^k)p^k &= -F(z^k), \\ z^{k+1} &:= z^k + p^k. \end{aligned} \tag{3.1}$$

Note that in this chapter, we denote the iterates by $\{z^k\}$ instead of $\{x_k\}$ to remain consistent with the notation used in the original papers of the methods discussed in the upcoming sections.

Newton's method is also often stated and used as a method for minimization problems

$$\min_{x \in \mathbb{R}^n} f(x)$$

with $f\colon \mathbb{R}^n \to \mathbb{R}$ by setting $F = \nabla f$. This makes the version for systems of equations the more general option as not every function occurs as the gradient of another.

The following is a well known convergence result for Newton's method.

Theorem 3.1. *Let F be differentiable and let the Jacobian F' be locally Lipschitz continuous in a neighborhood of a solution z^* at which $F'(z^*)$ is invertible. Then, there exists an $r > 0$ such that for all $z^0 \in B_r(z^*)$, the sequence $\{z^k\}$ given by (3.1) converges to z^*. The convergence rate is quadratic and additionally $\{F(z^k)\}$ also converges quadratically towards zero.*

For a proof, see for example [33, Satz 5.26]. The fast quadratic convergence rate is the reason behind the popularity of Newton's Method. However, this benefit comes with substantial downsides which are prohibitive for many applications. For one, the convergence result makes strong smoothness assumptions on F, and $F'(z^*)$ may not even be invertible. In general, the solution z^* need not be isolated, which is always the case when $F'(z^*)$ is

invertible, see Theorem 3.3. The other hurdle is the high computational complexity. In every step we need to solve a linear system. In fact, calculating the derivative may already be too expensive or even infeasible as the Jacobian may not fit into memory for large n. This makes Newton's method ill-suited for large scale systems. Finally, Newton's method only works for unconstrained systems, that is, we cannot guarantee that the iterates remain within the set Ω.

Thus, the challenge is improving Newton's method by extending it to more general classes of functions, lowering the computational complexity in each iteration and all this while maintaining a fast convergence rate. These goals are conflicting, since roughly speaking, when we go to more general functions, there are less properties we can exploit.

One big branch of Newton methods that address the complexity of the iterations are Quasi-Newton methods. Instead of using the Jacobian, they use some approximation that can be cheaply calculated or easily inverted. Often, the solution of the linear system can be calculated matrix-free.

To address the smoothness assumptions on F, there a various ways to define a generalized derivative, see Section 2.3. With this, we can define a Newton iteration by

$$Vp^k = -F(z^k), \quad V \in \partial F(z^k).$$

Indeed, when all the matrices $V \in \partial F(z^*)$ are invertible at a solution $z^* \in Z$ and z^0 is close enough to z^*, then we get superlinear convergence if F is semismooth, and even quadratic convergence if F is strongly semismooth [65, Theorem 3.2].

One particular discovery to generalize the regularity assumption at the solution is that the convergence speed rather depends on the *error bound condition* instead of the regularity.

Definition 3.2. A function $F\colon \mathbb{R}^n \to \mathbb{R}^m$ provides a local *error bound* around $z^* \in Z$, if there exist $\ell, \delta > 0$, such that

$$\operatorname{dist}[s, Z] \leq \ell \|F(s)\|$$

for all $s \in B_\delta(z^*)$.

Whenever $F'(z^*)$ is invertible, not only is z^* an isolated solution, F also provides a local error bound around z^*.

Theorem 3.3. *(cf. [74, Lemma 10.4]) Let $F\colon \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable around a solution $z^* \in Z$. When $F'(z^*)$ is non-singular, we have*

a) z^* is an isolated solution and
b) F provides a local error bound around z^*.

Proof. By assumption, there exists an $\varepsilon_0 > 0$ such that F' is continuous on $B_{\varepsilon_0}(z^*)$. As $F'(z^*)$ is invertible, we can define $\ell := 2\|F'(z^*)^{-1}\|$. By the definition of differentiability, there exists an $\varepsilon_1 \in (0, \varepsilon_0)$, such that

$$\|F(z) - F(z^*) - F'(z^*)(z - z^*)\| \leq \ell^{-1}\|z - z^*\|$$

holds for all $z \in B_{\varepsilon_1}(z^*)$. This implies

$$\begin{aligned} \|z - z^*\| &= 2\|F'(z^*)^{-1}F'(z^*)(z - z^*)\| - \|z - z^*\| \\ &\leq \ell\|F'(z^*)(z - z^*)\| - \ell\|F(z) - F(z^*) - F'(z^*)(z - z^*)\| \\ &\leq \ell\|F(z) - F(z^*)\| \\ &= \ell\|F(z)\| \end{aligned} \tag{3.2}$$

for all $z \in B_{\varepsilon_1}(z^*)$. Now, by (3.2), $F(z) = 0$ implies $z = z^*$ for all $z \in B_{\varepsilon_1}$. Thus, z^* is the only solution in $B_{\varepsilon_1}(z^*)$ and we have shown a). Additionally, when we set $\varepsilon := \varepsilon_1/2$, we get that $\operatorname{dist}[z, Z] = \|z - z^*\|$, which implies b) by (3.2), finalizing the proof. □

There are F that provide a local error bound around non-isolated solutions, e.g., $F\colon \mathbb{R}^2 \to \mathbb{R}, F(z) = z_1$ at all $z^* \in Z$. Section 3.1 discusses the error bound condition and its influence on the discussed method in that section.

3.1 LP-Newton

The LP-Newton method introduced in [24] encompasses the notions of generalized derivatives and the error bound condition while still maintaining quadratic convergence. The cost for achieving this greater generality is that we need to solve a Linear Program as a subproblem instead of a linear system in each iteration. However, we also get the benefit that the method works on constrained systems (1.1) and it is one of the few which allow $m \neq n$. Coming form an iterate $s \in \mathbb{R}^n$, the next iterate is generated as a solution to the following subproblem

$$\begin{aligned} \min_{\gamma, z} \gamma \quad \text{s.t. } & z \in \Omega, \\ & \|F(s) + G(s)(z - s)\| \leq \gamma \|F(s)\|^2, \\ & \|z - s\| \leq \gamma \|F(s)\|, \\ & \gamma \geq 0. \end{aligned} \tag{3.3}$$

Here, the mapping $G\colon \mathbb{R}^n \to \mathbb{R}^{m \times n}$ is the replacement for the Jacobian. LP-Newton does not mandate which derivative to use, as long as its assumptions are satisfied. When $\|\cdot\|$ is the infinity norm and Ω is polyhedral, this is a Linear Program. For instance, if $\Omega = \{z \in \mathbb{R}^n \mid Az \leq b\}$ with a matrix $A \in \mathbb{R}^{r \times n}$ and $b \in \mathbb{R}^r$, then (3.3) becomes

$$\begin{aligned} \min_{\gamma, z} \gamma \quad \text{s.t. } & Az \leq b, \\ & G(s)z \leq \gamma \|F(s)\|^2 \cdot \mathbf{1} - F(s) + G(s)s, \\ & -G(s)z \leq \gamma \|F(s)\|^2 \cdot \mathbf{1} + F(s) - G(s)s, \\ & z \leq \gamma \|F(s)\| \cdot \mathbf{1} + s, \\ & -z \leq \gamma \|F(s)\| \cdot \mathbf{1} - s, \\ & \gamma \geq 0, \end{aligned}$$

with $\mathbf{1} := (1, \ldots, 1)^\top \in \mathbb{R}^n$. That is indeed a Linear Program.

From this, we get the following algorithm.

Algorithm 3.1 LP-Newton

1: Choose $z^0 \in \Omega$ and a mapping $G\colon \mathbb{R}^n \to \mathbb{R}^{m \times n}$, $k \leftarrow 0$.
2: If $z^k \in Z$ then stop.
3: Compute a solution (γ^{k+1}, z^{k+1}) of (3.3) with $s := z^k$.
4: $k \leftarrow k + 1$, go to **Step 2**

This algorithm is well-defined by the following proposition.

Proposition 3.4. *(see [24, Proposition 1]) For any $s \in \mathbb{R}^n$,*

a) problem (3.3) *has a solution and*
b) the optimal value of (3.3) *is zero if and only if $s \in Z$*

Next, we will look at the assumptions that guarantee the convergence properties of Algorithm 3.1. For this, we fix a solution $z^* \in Z$ and a radius $\delta > 0$. All assumptions shall hold on the ball $B_\delta(z^*)$.

The first assumption is a Lipschitz-like assumption which is strictly weaker than local Lipschitz-continuity at z^*.

Assumption 1. There exists $L > 0$ such that

$$\|F(s)\| \leq L \operatorname{dist}[s, Z]$$

for all $s \in B_\delta(z^*) \cap \Omega$.

The second assumption is a generalization of the aforementioned error bound condition. Here, we deviate from the original paper, as we introduce an additional parameter σ to emphasize the impact on the convergence behavior. This generalization of the error bound condition goes as follows

Assumption 2. There exists $\ell > 0$ and $\sigma > 0$ such that

$$\operatorname{dist}[s, Z] \leq \ell \|F(s)\|^\sigma$$

for all $s \in B_\delta(z^*) \cap \Omega$.

Here, $\sigma = 1$ yields the default error bound condition, while $\sigma > 1$ is more demanding and $\sigma < 1$ is less demanding than the error bound condition. As we will see, this assumption is the primary factor in the convergence rate of Algorithm 3.1, and the degree to which the error bound condition holds directly influences it. In fact, the convergence rate of $\{z^k\}$ will be linear in σ and we will also achieve convergence in cases where $\sigma < 1$.

The next two assumptions are what restricts our choices for the mapping G.

Assumption 3. There exists $\Gamma \geq 1$ such that

$$\gamma(s) \leq \Gamma$$

for all $s \in B_\delta(z^*) \cap \Omega$.

Assumption 4. There exists $\hat{\alpha} > 0$ such that

$$w \in \mathscr{L}(s, \alpha) := \left\{ w \in \Omega \,|\, \|w - s\| \leq \alpha, \|F(s) + G(s)(w - s)\| \leq \alpha^2 \right\}$$

implies

$$\|F(w)\| \leq \hat{\alpha} \alpha^2$$

for all $s \in (B_\delta(z^*) \cap \Omega) \setminus Z$ and all $\alpha \in [0, \delta]$.

This assumption specifies how well $F(s) + G(s)(\cdot - s)$ needs to approximate F'. To give an example of when Assumptions 3 and 4 hold, consider the following theorem.

Theorem 3.5. *(cf. [24, Corollary 2]) Let F be locally Lipschitz continuous and assume there exists a $\kappa > 0$ such that*

$$\sup\{\|F(s) + V(z^* - s)\| : V \in \partial F(s)\} \leq \kappa \|z^* - s\|^2 \tag{3.4}$$

for all $s \in B_\delta(z^*) \cap \Omega$. When all the matrices in $\partial_B F(z^*)$ have rank equal to n and G is chosen such that $G(s) \in \partial_B F(s)$ for all $s \in B_\delta(z^*) \cap \Omega$, then Assumptions 3 and 4 hold for δ sufficiently small.

The condition (3.4) also occurs in the LP-Newton paper [24] as Condition 2. This condition holds whenever F is strongly semismooth at the solution z^* [27, Lemma 17]. Thus, F being locally Lipschitz, strongly semismooth at z^* and V having rank n for all $V \in \partial_B F(z^*)$, is sufficient for Assumptions 1, 3 and 4 when choosing $G(s) \in \partial_B F(s)$. For further discussion on the assumptions, we refer to [24]

Now, we will look at the convergence result and the needed lemmas. We will only provide proofs when they deviate from the original paper due to our modification of Assumption 2. Even with these modification, the proofs are however very similar.

Lemma 3.6. *(see [24, Lemma 1]) Let Assumption 3 be satisfied and define the set $\mathscr{F}(s, \Gamma)$ by*

$$\mathscr{F}(s, \Gamma) := \left\{ z \in \Omega \,|\, \|z - s\| \leq \Gamma \|F(s)\|, \|F(s) + G(s)(z - s)\| \leq \Gamma \|F(s)\|^2 \right\}.$$

Then, for any $s \in B_\delta(z^*) \cap \Omega$, the set $\mathscr{F}(s, \Gamma)$ is nonempty. If, in addition, Assumption 1 is satisfied, then

$$\|F(s) + G(s)(z - s)\| \leq L\Gamma^2 \operatorname{dist}[s, Z]^2 \quad \text{and} \quad \|z - s\| \leq L\Gamma \operatorname{dist}[s, Z]$$

hold for all $z \in \mathscr{F}(s, \Gamma)$.

Lemma 3.7. *(cf. [24, Lemma 2]) Let Assumptions 1–4 be satisfied and let $\sigma > \frac{1}{2}$. Then, there are $\epsilon > 0$ and $C > 0$ such that, for any $s \in B_\epsilon(z^*) \cap \Omega$,*

$$\operatorname{dist}[z, Z] \leq C \operatorname{dist}[s, Z]^{2\sigma} \leq \frac{1}{2} \operatorname{dist}[s, Z]$$

holds for all $z \in \mathscr{F}(s, \Gamma)$.

Proof. Let us choose $C := \hat{\alpha}^\sigma \ell \Gamma^{2\sigma} L^{2\sigma}$ and any ε according to

$$0 < \varepsilon \leq \min\left\{ \frac{\delta}{2}, \frac{\delta}{2} \Gamma^{-1} L^{-1}, (2C)^{-1/(2\sigma - 1)} \right\}. \tag{3.5}$$

Therefore

$$\|z^* - z\| \leq \|z^* - s\| + \|z - s\| \leq \frac{\delta}{2} + \frac{\delta}{2} = \delta,$$

which implies $z \in B_\delta(z^*) \cap \Omega$. Since $z \in \mathscr{F}(s, \Gamma)$ and $\Gamma \geq 1$ yield

$$\|F(s) + G(s)(z - s)\| \leq \Gamma \|F(s)\|^2 \leq \Gamma^2 \|F(s)\|^2 \quad \text{and} \quad \|z - s\| \leq \Gamma \|F(s)\|,$$

we have $z \in \mathscr{L}(s, \alpha)$, with $\alpha \coloneqq \Gamma\|F(s)\|$. Moreover,

$$\alpha = \Gamma\|F(s)\| \leq \Gamma L \operatorname{dist}[s, Z] \leq \Gamma L\varepsilon \leq \frac{\delta}{2}$$

follows by Assumption 1 and (3.5). Thus, Assumption 4 implies

$$\|F(z)\| \leq \hat{\alpha}\alpha^2 = \hat{\alpha}\Gamma^2\|F(s)\|^2.$$

Using this, Assumptions 1 and 2, and (3.5), we obtain

$$\begin{aligned}\operatorname{dist}[z, Z] &\leq \ell\|F(z)\|^\sigma \leq \hat{\alpha}^\sigma \ell \Gamma^{2\sigma}\|F(s)\|^{2\sigma} \\ &\leq C \operatorname{dist}[s, Z]^{2\sigma} \leq C\varepsilon^{2\sigma-1} \operatorname{dist}[s, Z] \leq \frac{1}{2}\operatorname{dist}[s, Z]\end{aligned}$$

and the assertion follows. □

Now, we state the actual convergence result. We modified the theorem to more general sequences $\{z^k\}$ to shorten the proof to the parts that change due to our modification of Assumption 2.

Theorem 3.8. *(cf. [24, Theorem 1]) Let Assumptions 1–4 be satisfied and let $\{z^k\}$ be any sequence such that*

$$z^k \in B_\varepsilon(z^*) \tag{3.6}$$

with ε according to (3.5) and

$$z^{k+1} \in \mathscr{F}(z^k, \Gamma) \tag{3.7}$$

hold for all $k \in \mathbb{N}_0$. Then, $\{z^k\}$ converges to some $\hat{z} \in Z$ with convergence order 2σ.

Proof. Because of (3.6) and (3.7), Lemma 3.7 provides

$$\operatorname{dist}[z^{k+1}, Z] \leq C \operatorname{dist}[z^k, Z]^{2\sigma} \leq \frac{1}{2}\operatorname{dist}[z^k, Z] \tag{3.8}$$

for all $k \in \mathbb{N}_0$. This yields

$$\lim_{k\to\infty} \operatorname{dist}[z^k, Z] = 0. \tag{3.9}$$

Due to (3.7), we can apply Lemma 3.6, and then (3.8) to obtain for all $j, k \in \mathbb{N}_0$ with $k > j$ that

$$\|z^k - z^j\| \leq \sum_{i=j}^{k-1} \|z^{i+1} - z^i\| \leq \Gamma L \operatorname{dist}[z^j, Z] \sum_{i=j}^{k-1} \left(\frac{1}{2}\right)^{i-j} \leq 2\Gamma L \operatorname{dist}[z^j, Z]. \tag{3.10}$$

Because the right hand term tends to zero by (3.9), $\{z^k\}$ is a Cauchy sequence and thus converges to some $\hat{z} \in \mathbb{R}^n$. By closedness of Z, and (3.9), we additionally have $\hat{z} \in Z$. Finally, we can use (3.10) and (3.8) to get

$$\|z^{k+j} - z^{k+1}\| \leq 2\Gamma L \operatorname{dist}[z^{k+1}, Z] \leq 2C\Gamma L \operatorname{dist}[z^k, Z]^{2\sigma} \leq 2C\Gamma L\|\hat{z} - z^k\|^{2\sigma},$$

where $j \to \infty$ leads to

$$\|\hat{z} - z^{k+1}\| \leq 2C\Gamma L \operatorname{dist}[z^k, Z]^{2\sigma} \leq 2C\Gamma L\|\hat{z} - z^k\|^{2\sigma}, \tag{3.11}$$

proving the desired convergence order of 2σ. □

And indeed, any sequence generated by Algorithm 3.1 with z^0 close enough to z^* satisfies the assumptions (3.6) and (3.7), see [24, Proof of Theorem 1]. Our modification of Assumption 2 does not interfere with the proof in any way. From this, it is also apparent that the algorithm does not depend on an exact solution of subproblem (3.3). A close enough approximation in each step suffices to achieve the desired convergence order, simplifying the implementation.

Finally, we will briefly discuss the convergence order of the residuals to point out the changes that arise from our modification. In the default case $\sigma = 1$, both the iterates and residuals converge quadratically, i.e., the convergence orders are identical. However, in the general case we obtain with Assumptions 1 and 2 and (3.11) that

$$\begin{aligned}\|F(z^{k+1})\| &\leq L \operatorname{dist}[z^{k+1}, Z] \leq L\|z^{k+1} - \hat{z}\| \\ &\leq 2C\Gamma L^2 \operatorname{dist}[z^k, Z]^{2\sigma} \leq 2C\Gamma L^2 \ell^{2\sigma} \|F(z^k)\|^{2\sigma^2},\end{aligned}$$

showing that $\{F(z^k)\}$ converges to zero with convergence order $2\sigma^2$ instead of 2σ. Thus, they coincide only for $\sigma = 1$. For $\sigma > 1$, the residuals converge faster towards zero than the iterates, while for $\sigma < 1$ they converge slower.

3.2 Secant Modified LP-Newton

While LP-Newton greatly relaxes the assumptions on the function F, it does not lower the computational complexity but rather increases it. One method which tries to address this problem is the *Secant Modified LP-Newton* (SMLP-Newton) [53]. It is an extension of the authors' previous work [52] which extends the LP-Newton method by incorporating Quasi-Newton approximations of the Jacobian. However, in [52] they only achieved linear convergence. For SMLP-Newton, they improved this result to superlinear convergence.

In each step of LP-Newton, we employ a Quasi-Newton approximation M_k instead of $G(z^k)$ and we also replace one of the residuals $\|F(z^k)\|$ in (3.3) by a sequence $\{\eta_k\}$. With that, the new subproblem is given by

$$\begin{aligned}\min_{\gamma, z} \gamma \quad \text{s.t. } & z \in \Omega, \\ & \|F(z^k) + M_k(z - z^k)\| \leq \eta_k \gamma, \\ & \|z - z^k\| \leq \gamma.\end{aligned} \tag{3.12}$$

Here, in both inequalities, the authors omitted a $\|F(z^k)\|$ term as it can be absorbed into γ. Note that $\|z - z^k\| \leq \gamma$ automatically guarantees $\gamma \geq 0$. The matrix M_k is updated in each step to the closest matrix that satisfies the *secant equation*

$$M_{k+1}(z^{k+1} - z^k) = F(z^{k+1}) - F(z^k),$$

which is a very common tool in Quasi-Newton methods, see e.g., [20, 56]. The motivation behind this equation is that the true derivative provides the approximation

$$F'(z^k)(z^{k+1} - z^k) \approx F(z^{k+1}) - F(z^k)$$

by Taylor's Theorem. We also assert $M_{k+1} \in \mathcal{X}$ for a fixed closed and convex set $\mathcal{X} \subseteq \mathbb{R}^{m\times n}$. Hence, M_{k+1} is the solution to the problem

$$\begin{aligned} \min_N \|N - M_k\|_\star^2 \;\text{ s.t. }\; & N(z^{k+1} - z^k) = F(z^{k+1}) - F(z^k), \\ & N \in \mathcal{X}, \end{aligned} \tag{3.13}$$

where $\|\cdot\|_\star$ is some norm on $\mathbb{R}^{m\times n}$ induced by an inner product. Similarly, $B_\delta^\star(M)$ will denote the δ-neighborhood of M wrt. $\|\cdot\|$. The set $\mathcal{X}$ can be chosen to include a-priori knowledge on the Jacobians that can occur for a given problem. If for example all the derivatives are known to be symmetric, $\mathcal{X}$ can be chosen as the set of all symmetric matrices. It is also important to note that (3.13) may not have a solution for poor choices of $\mathcal{X}$. Thus, $\mathcal{X}$ has to be chosen with some care. Two important choices for which the existence of a solution is ensured are $\mathcal{X} = \mathbb{R}^{m\times n}$ and the space of the symmetric matrices. When $\|\cdot\|_\star$ is the Frobienius Norm, M_{k+1} is then given by the Broyden Update in the first case (see [20, Lemma 8.1.1]), and as the Powell-symmetric-Bryoden update in the second case (see [32, Satz 11.3]).

Finally, η_k is updated according to

$$\eta_{k+1} = \min\Big\{\eta_0, \kappa \max\Big\{\|F(z^k)\|^\sigma, \|F(z^{k+1})\|^\sigma\Big\}\Big\}, \tag{3.14}$$

with $\eta_0 > 0$ and a fixed parameter $0 < \sigma \leq 1$. In the original Quasi-Newton modification of LP-Newton in [52], the authors used a constant parameter instead of updating η_k in each iteration. In the proof that $F(z^k) \to 0$, it is also shown that $\eta_k \to 0$. This is responsible for the superlinear convergence of SMLP-Newton, whereas a constant η only yields linear convergence in [52]

Algorithm 3.2 Secant Modified LP-Newton (SMLP)

1: Choose $z^0 \in \Omega$, $\eta_0 \in \mathbb{R}_{>0}$, $M_0 \in \mathbb{R}^{m\times n}$ and set $k \leftarrow 0$.
2: If $z^k \in Z$ then stop.
3: Compute a solution (γ^{k+1}, z^{k+1}) of (3.12).
4: Compute M_{k+1} according to (3.13) and η_{k+1} according to (3.14).
5: Set $k \leftarrow k+1$, go to **Step 2**

For the assumptions of SMLP, fix a radius $\varepsilon_0 > 0$. The smoothness assumptions for SMLP are stronger than those for LP-Newton. Specifically, we need again that F is differentiable and that both F and F' are Lipschitz in a fixed solution $z^* \in Z$.

Assumption 1. There exist $L_0, L_1 > 0$ such that

$$\|F(z) - F(w)\| \leq L_0\|z - w\| \quad \text{and} \quad \|F'(z) - F'(w)\| \leq L_1\|z - w\|$$

for all $z, w \in B_{2\varepsilon_0}(z^*)$.

Here, $\|\cdot\|$ on $\mathbb{R}^{m\times n}$ is the operator norm (see (2.1)). Concerning regularity, we can again work with the error bound condition.

Assumption 2. There exists $\ell > 0$ such that

$$\operatorname{dist}[s, Z] \leq \ell \|F(s)\|$$

for all $s \in B_{\varepsilon_0}(z^*) \cap \Omega$.

The last assumption requires that the Quasi-Newton approximations M_k are close enough to the real derivatives of F along the iterates. However, authors acknowledge that it may not hold in practice. We will discuss this further in the numerics chapter.

Assumption 3. There exists $c > 0$ such that

$$\|N_{k+1} - M_{k+1}\| \leq c\|z^{k+1} - z^k\|^\sigma$$

for all $k \in \mathbb{N}_0$, where N_{k+1} is the average Jacobian of F between z^k and z^{k+1} defined by

$$N_{k+1} = \int_0^1 F'(z^k + t(z^{k+1} - z^k))\,\mathrm{d}t.$$

Now, we will state the following lemma which is necessary in proving superlinear convergence of SMLP-Newton. We include it here, as we will also need it to prove a slightly stronger result afterwards.

Lemma 3.9. *Let Assumptions 1–3 be satisfied and $\{z^k\}, \{M_k\}$ and $\{\eta_k\}$ be generated by Algorithm 3.2 with $\kappa \geq (c + L_1)/\ell^\sigma$. Then, there exist $r, \varepsilon, \delta > 0$ such that if*

$$\eta_0 < \frac{1}{6\ell}, \quad z^0 \in B_r(z^*) \cap \Omega \quad \text{and} \quad M_0 \in B^\star_{\delta/2}(F'(z^*)) \cap \mathcal{X},$$

then for all $k \in \mathbb{N}_0$,

(i) $\|z^k - z^*\| \leq \varepsilon$,
(ii) $\|M_k - F'(z^*)\|_\star \leq \delta$,
(iii) $\|F(z^k) + M_k(z^{k+1} - z^k)\| \leq \eta_k \operatorname{dist}[z^k, Z]$,
(iv) $\|z^{k+1} - z^k\| \leq \operatorname{dist}[z^k, Z]$,
(v) $\operatorname{dist}[z^{k+1}, Z] \leq \frac{1}{2}\operatorname{dist}[z^k, Z]$.

The proof can be again be found in the original paper [53, Lemma 4.1], as well as the actual proof for the superlinear convergence of $\{F(z^k)\}$ and $\{z^k\}$ (Theorem 4.1 and Corollary 4.1). Important to note is that the assumptions of Lemma 3.9 are exactly the assumptions needed for these theorems.

Now, we will prove an additional result that substantiates the superlinear convergence proven in [53]. We will however assume that the superlinear convergence of $\{F(z^k)\}$ and $\{z^k\}$ is already given by the mentioned theorems, that is $\{F(z^k)\}$ converges superlinearly to zero and $\{z^k\}$ converges superlinearly to some solution $\hat{z}$.

Theorem 3.10. *Under the assumptions of Lemma 3.9, there exist $C_0 > 0$ and $C_1 > 0$ such that*

$$\|F(z^{k+1})\| \leq C_0\|F(z^{k-1})\|^\sigma\|F(z^k)\| \quad \text{and} \quad \|z^{k+1} - \hat{z}\| \leq C_1\|z^{k-1} - \hat{z}\|^\sigma\|z^k - \hat{z}\|$$

for large enough k.

Proof. First, we want to get an upper bound for the difference $\|M_{k+1} - M_k\|$ of our Quasi-Newton approximations in terms of the residuals. We start off by

$$\begin{aligned}
\|N_{k+1} - N_k\| &\leq \int_0^1 \left\| F'\left(z^k + t(z^{k+1} - z^k)\right) - F'\left(z^{k-1} + t(z^k - z^{k-1})\right) \right\| \mathrm{d}t \\
&\leq \int_0^1 L_1 \left\| z^k - z^{k-1} + t(z^{k+1} - z^k - z^k + z^{k-1}) \right\| \mathrm{d}t \\
&\leq L_1 \|z^k - z^{k+1}\| + \frac{L_1}{2}\left(\|z^{k+1} - z^k\| + \|z^k - z^{k-1}\|\right) \\
&\leq \frac{3L_1}{2} \operatorname{dist}[z^k, Z] + \frac{L_1}{2}\|z^k - z^{k-1}\| \\
&\leq \frac{3L_1}{4} \operatorname{dist}[z^{k-1}, Z] + \frac{L_1}{2} \operatorname{dist}[z^{k-1}, Z] \\
&= \frac{5L_1}{4} \operatorname{dist}[z^{k-1}, Z],
\end{aligned}$$

where we used Assumption 1, Lemma 3.9 (iv) and the fact that $\operatorname{dist}[z^k, Z] < 1$ for sufficiently large k. With that and Assumptions 2 and 3, we get

$$\begin{aligned}
\|M_{k+1} - M_k\| &\leq \|M_{k+1} - N_{k+1}\| + \|N_{k+1} - N_k\| + \|N_k - M_k\| \\
&\leq c\|z^{k+1} - z^k\|^\sigma + \frac{5L_1}{4} \operatorname{dist}[z^{k-1}, Z] + c\|z^k - z^{k-1}\|^\sigma \\
&\leq c \operatorname{dist}[z^k, Z]^\sigma + \frac{5L_1}{4} \operatorname{dist}[z^{k-1}, Z]^\sigma + c \operatorname{dist}[z^{k-1}, Z]^\sigma \\
&\leq \frac{6c + 5L_1}{4} \operatorname{dist}[z^{k-1}, Z]^\sigma \\
&\leq C\|F(z^{k-1})\|^\sigma
\end{aligned}$$

with $C = \ell\,(6c + 5L_1)\,/4$. Finally, since $\|F(z^k)\|$ converges linearly, it holds that $\|F(z^{k+1})\| < \|F(z^k)\|$ for sufficiently large k. Therefore, $\eta_k = \kappa\|F(z^{k-1})\|^\sigma$ for sufficiently large k and we have

$$\begin{aligned}
\|F(z^{k+1})\| &= \|F(z^k) + M_{k+1}(z^{k+1} - z^k)\| \\
&\leq (\eta_k + \|M_{k+1} - M_k\|) \operatorname{dist}[z^k, Z] \\
&\leq (\kappa\|F(z^{k-1})\|^\sigma + C\|F(z^{k-1})\|^\sigma) \operatorname{dist}[z^k, Z] \\
&\leq (\kappa + C)\ell\|F(z^{k-1})\|^\sigma \|F(z^k)\|.
\end{aligned}$$

proving the first inequality with $C_0 = (\kappa + C_0)\ell$. For the second inequality, first note that by Lemma 3.9 (iv) and (v)

$$\begin{aligned}
\|z^{k+j} - z^k\| &\leq \sum_{i=k}^{k+j-1} \|z^{i+1} - z^i\| \\
&\leq \sum_{i=k}^{k+j-1} \operatorname{dist}[z^i, Z] \\
&\leq \sum_{i=k}^{k+j-1} \frac{1}{2^{i-k}} \operatorname{dist}[z^k, Z] \\
&\leq 2 \operatorname{dist}[z^k, Z]
\end{aligned}$$

holds, where $j \to \infty$ yields

$$\|z^k - \hat{z}\| \leq 2\,\mathrm{dist}[z^k, Z].$$

The original authors also used these inequalities to prove the convergence of $\{z^k\}$. Finally, with this, Assumptions 1 and 2 and the already proven inequality, we get

$$\begin{aligned}\|z^{k+1} - \hat{z}\| &\leq 2\,\mathrm{dist}[z^{k+1}, Z] \\ &\leq 2\ell\|F(z^{k+1})\| \\ &\leq 2\ell C_0\|F(z^{k-1})\|^\sigma\|F(z^k)\| \\ &\leq 2L_0^2\ell C_0\|z^{k-1} - \hat{z}\|^\sigma\|z^k - \hat{z}\|,\end{aligned}$$

also proving the second inequality with $C_1 = 2L_0^2\ell C_0$. □

Note that Theorem 3.10 is stronger than the superlinear convergence of $\{F(z^k)\}$ and of $\{z^k\}$. We see that the next iteration is not just bounded by the current iteration, but we also observe an additional "echoing" of the previous iteration with power σ. In addition, we can derive a convergence order of $1 + \sigma$ over two steps by $\|F(z^k)\| \leq \|F(z^{k-1})\|$, that is

$$\limsup_{k\to\infty} \frac{\|F(z^{k+2})\|}{\|F(z^k)\|^p} < \infty \tag{3.15}$$

for $p = 1 + \sigma$. If we also had that the residuals $\|F(z^{k-1})\|$ were in some way bounded by the next iteration $\|F(z^k)\|$, limiting how much we can improve in each step, we even would get an actual convergence order of $1 + \sigma$. The same results of course hold also true for the iterates $\{z^k\}$, as

$$\frac{\|z^{k+2} - \hat{z}\|}{\|z^k - \hat{z}\|^p} \leq C_1 \frac{\|z^k - \hat{z}\|^\sigma\|z^{k+1} - \hat{z}\|}{\|z^k - \hat{z}\|^{1+\sigma}} \leq C_1^2\|z^{k-1} - \hat{z}\|^\sigma \longrightarrow 0$$

for $p = 1 + \sigma$ and large enough k.

4

Conjugate Gradient Methods

The traditional Conjugate Gradient (CG) Method was originally studied as a method to find solutions to the linear equation $Ax = b$, where A is symmetric and positive definite. In particular, the method finds the solution to the strictly convex optimization problem

$$\min_x \frac{1}{2} x^\top A x - b^\top x, \tag{4.1}$$

which coincides with the solution of $Ax = b$ by the optimality condition. A naïve way to solve this problem could be gradient descent, where we use the negative gradient as a search direction, as it is guaranteed to be the direction of steepest descent. While this converges, it does so slowly. The trick behind the Conjugate Gradient Method is that we choose the search directions d_k to be *conjugate* to each other wrt. A, that is $d_i^\top A d_j = 0$ for $i \neq j$. The iterates x_k are then given as usual by $x_{k+1} = x_k + \alpha_k d_k$). If we additionally choose the step length α_k such that it minimizes $f(x_k + \alpha_k d_k)$, then the iterates yield the *expanding subspace minimization* property, that is x_k minimizes f on

$$x_0 + \operatorname{span}(d_0, \dots, d_{k-1}),$$

see [56, Theorem 5.2.]. Since the d_k are linearly independent, the algorithm thus terminates in at most n steps. Fortunately, there is a simple way to ensure all the search directions are conjugate by modifying the gradient descent direction by a multiple of the previous search direction, i.e.,

$$d_k = -\nabla f_k + \beta_k d_{k-1}$$

with a scalar $\beta_k \in \mathbb{R}$ and $\nabla f_k \coloneqq \nabla f(x_k)$. By ensuring d_{k-1} and d_k to be conjugate and by applying $d_{k-1}^\top A$, we deduce

$$\beta_k = \frac{\nabla f_k^\top A d_{k-1}}{d_{k-1}^\top A d_{k-1}}.$$

In fact, with this choice of β_k and d_0 chosen as the exact gradient descent direction (and only then), the d_k are automatically conjugate [56, Theorem 5.3.]. Additionally, α_k from above is explicitly given by

$$\alpha_k = \frac{\nabla f_k^\top d_k}{d_k^\top A d_k}.$$

The gradient $\nabla f(x_k)$ can also be iteratively calculated via

$$\nabla f_{k+1} = \nabla f_k + \alpha_k A d_k.$$

Thus in each step, the only possibly expensive operation is the calculation of Ad_k. All other computations are performed in linear-time and independently of A. This means the efficiency of the algorithm entirely depends on the efficiency of the evaluation of $x \mapsto Ax$, which depending on A could be implemented potentially matrix-free. This is particularly the case when A is sparse. This makes the CG method very efficient speed- and memory-wise.

As the speed and efficiency of the CG method are very desirable properties, mathematicians have generalized it to more general functions, the first being Fletcher and Reeves [30]. Trying to generalize the CG parameter leads to various possible choices, as we have already described in Section 1.1. For the convenience of the reader, we again include an overview here in Table 4.1. All of these choices coincide in the quadratic case.

Coefficient	Source
$\beta_{k+1}^{\mathrm{HS}} = \dfrac{\nabla f_{k+1}^\top y_k}{y_k^\top d_k}$	Hestenes and Stiefel [36]
$\beta_{k+1}^{\mathrm{FR}} = \dfrac{\nabla f_{k+1}^\top \nabla f_{k+1}}{\nabla f_k^\top \nabla f_k}$	Fletcher and Reeves [30]
$\beta_{k+1}^{\mathrm{PRP}} = \dfrac{\nabla f_{k+1}^\top y_k}{\nabla f_k^\top \nabla f_k}$	Polak and Ribière [61] and Polyak [62]
$\beta_{k+1}^{\mathrm{CD}} = \dfrac{\nabla f_{k+1}^\top \nabla f_{k+1}}{-\nabla f_k^\top d_k}$	Fletcher [29] (Conjugate Descent)
$\beta_{k+1}^{\mathrm{LS}} = \dfrac{\nabla f_{k+1}^\top y_k}{-\nabla f_{k+1}^\top d_k}$	Liu and Storey [48]
$\beta_{k+1}^{\mathrm{DY}} = \dfrac{\nabla f_{k+1}^\top \nabla f_{k+1}}{y_k^\top d_k}$	Dai and Yuan [18]
$\beta_{k+1}^{\mathrm{DL}} = \dfrac{\nabla f_{k+1}^\top y_k}{y_k \top d_k} - t\dfrac{\nabla f_{k+1}^\top s_k}{y_k \top d_k}$	Dai and Liao [17]
$\beta_{k+1}^{\mathrm{HZ}} = \dfrac{\nabla f_{k+1}^\top}{y_k^\top d_k}\left(y_k - 2\dfrac{y_k^\top y_k}{y_k^\top d_k} d_k\right)$	Hager and Zhang [34]

Table 4.1: Various common choices for β_k with the common abbreviations $y_k \coloneqq \nabla f_{k+1} - \nabla f_k$ and $s_k \coloneqq x_{k+1} - x_k$. The second-to-last coefficient $\beta_{k+1}^{\mathrm{DL}}$ contains a parameter $t > 0$, whose optimal choice remains a subject of research [3].

The necessary assumptions for these CG methods are that f must be convex and continuously differentiable. In that case, x^* being a minimizer of f is equivalent to $\nabla f(x^*) = 0$. That is, similarly to Newton's method, we can also view the CG method as a method to solve that equation and try to generalize it to continuous functions F that are not the gradient of another function. As f must be convex, we first need to find a property of ∇f that encompasses convexity, that we can then generalize to F.

Definition 4.1. (cf. [32, Definition 3.6]) A function $F\colon \mathbb{R}^n \to \mathbb{R}^n$ is called *monotonically increasing* if

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

Indeed, this is the property we need.

Lemma 4.2. *(cf. [32, Satz 3.7]) Let $S \subseteq \mathbb{R}^n$ be convex and $f\colon S \to \mathbb{R}$ be continuously differentiable. Then*

$$f \text{ is convex} \iff \nabla f \text{ is monotonically increasing.}$$

Proof. For the sufficient condition, let f be convex and $x, y \in S$. Then

$$\begin{aligned}\langle \nabla f(x), y - x\rangle &= \lim_{h\to 0^+} \frac{f(x + h(y-x)) - f(x)}{h} \\ &\leq \lim_{h\to 0^+} \frac{hf(y) - hf(x)}{h} \\ &= f(y) - f(x).\end{aligned}$$

Similarly, we get

$$\langle \nabla f(y), x - y\rangle \leq f(x) - f(y).$$

Adding these two inequalities and rearranging the result leads to

$$\langle \nabla f(y) - \nabla f(x), y - x\rangle \geq 0.$$

For the necessary condition, let $x, y \in S$, $x \neq y$ and $c = (1-\tau)x + \tau y$ with $\tau \in (0,1)$. Then, by the mean value theorem, there are $\xi \in S$ between x and c, and $\eta \in S$ between c and y, such that

$$f(c) - f(x) = \langle \nabla f(\xi), c - x\rangle \quad \text{and} \quad f(y) - f(c) = \langle \nabla f(\eta), y - c\rangle, \tag{4.2}$$

Since ξ and η are on the line segment between x and y with η closer to y, there is an $s > 0$ such that $\eta - \xi = s(y - x)$. Now, we use the monotonicity of ∇f to get

$$0 \leq \langle \nabla f(\eta) - \nabla f(\xi), \eta - \xi\rangle = s\langle \nabla f(\eta) - \nabla f(\xi), y - x\rangle.$$

From this, we conclude $0 \leq \langle \nabla f(\eta) - \nabla f(\xi), y - x\rangle$. With (4.2) and $c - x = \tau(y - x)$ and $y - c = (1-\tau)(y-x)$, we finally get

$$\begin{aligned}0 &\leq \langle \nabla f(\eta) - \nabla f(\xi), y - x\rangle \\ &= \frac{1}{1-\tau}(f(y) - f(c)) - \frac{1}{\tau}(f(c) - f(x)),\end{aligned}$$

where multiplying $\tau(1-\tau)$ and then adding $f(c)$ yields

$$f((1-\tau)x + \tau y) \leq (1-\tau)f(x) + \tau f(y)$$

and thus the convexity of f. □

Notice that when F is monotonically *decreasing*, that is $\langle F(x) - F(y), x - y\rangle \leq 0$, we can just use $-F$ instead. Thus in practice, it does not matter whether or not F is monotonically increasing or decreasing and we will call F just *monotone* when it is actually monotonically *increasing*.

The general framework for CG methods on systems of equations is as follows. Let $F\colon \mathbb{R}^n \to \mathbb{R}^n$ be continuous and monotone and let $\Omega \subseteq \mathbb{R}^n$ be convex. Assume that the solution set Z is nonempty. In most cases, the search direction will be of the form

$$d_k = \begin{cases} -F(x_k) & \text{if} \quad k = 0 \\ -F(x_k) + \beta_k d_{k-1} & \text{if} \quad k > 0 \end{cases}$$

with a scalar $\beta_k \in \mathbb{R}$.

CG methods differ in their particular choice of β_k, their line-search strategy or possibly additional auxiliary steps. There are also other possible modification of d_{k+1}, like replacing $-F(x_k)$ with a scalar multiple. Other variants that add a third term that is also easily calculated also exist. For the purpose of this thesis, these *three-term methods* will also be regarded as CG methods.

4.1 The CG Method framework for constrained systems of equations

For the rest of this chapter, $F\colon \mathbb{R}^n \to \mathbb{R}^n$ will be a continuous monotone function. Also note that here, n equals m. Under these assumptions, the set Z is convex [58, Theorem 5.4.7]. Additionally, $\|\cdot\|$ will always denote the euclidean norm. Most recent CG methods for constrained systems of equations are of the following structure. For a sequence of iterates $\{x_k\}$, we use the abbreviation $F_k \coloneqq F(x_k)$. The general framework that most modern CG methods for constrained systems of equations abide by, is the following.

Algorithm 4.1 CG Method framework for constrained systems of equations

1: Choose $x_0 \in \mathbb{R}^n$, $\xi > 0$, $\rho \in (0,1)$ $d_0 \leftarrow -F(x_0)$, $k \leftarrow 0$.
2: If $\|F(x_k)\| \leq \varepsilon$ then stop.
3: Let $z_k = x_k + \alpha_k d_k$ with $\alpha_k = \xi\rho^m$ where m is the smallest $m \in \mathbb{N}_0$ such that the line search condition is fulfilled
4: If $z_k \in \Omega$ and $\|F(z_k)\| \leq \varepsilon$, then $x_{k+1} \coloneqq z_k$ and stop. Else, determine

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

where $0 < \gamma < 2$ and

$$\mu_k = \frac{F(z_k)^\top (x_k - z_k)}{\|F(z_k)\|^2}$$

5: Calculate d_{k+1}
6: $k \leftarrow k+1$, go to **Step 2**

Since we cannot explicitly calculate a suitable α_k, we need a backtracking line search strategy to to find such. The two most common line search conditions that CG methods choose from are

$$-F(x_k + \alpha_k d_k)^\top d_k \geq \kappa\alpha_k \|d_k\|^2, \tag{L1}$$
$$-F(x_k + \alpha_k d_k)^\top d_k \geq \kappa\alpha_k \|F(x_k + \alpha_k d_k)\| \|d_k\|^2. \tag{L2}$$

These will be the only line search conditions considered in this thesis.

Note that Algorithm 4.1 does not impose any particular form of d_k. The only condition that d_k must satisfy is the *decrease condition*

$$F_k^\top d_k < 0. \tag{4.3}$$

This condition guarantees, that the backtracking line search will eventually terminate. To prove this, assume that no value of α_k satisfies the backtracking line search condition. This implies that

$$-F(x_k + \xi\rho^m d_k)^\top d_k < \kappa\xi\rho^m \|d_k\|^2$$

for all $m \in \mathbb{N}_0$ in the case of line search condition (L1), and

$$-F(x_k + \xi\rho^m d_k)^\top d_k < \kappa\xi\rho^m \|F(x_k + \xi\rho^m d_k)\| \|d_k\|^2$$

for all $m \in \mathbb{N}_0$ in the case of line search condition (L2). In both of these cases, sending $m \to \infty$ yields

$$-F_k^\top d_k \leq 0$$

by the continuity of F. This is a contradiction to (4.3). Thus, the descent condition guarantees that eventually, either of the two chosen line search condition holds.

The most eminent difference to the original non-linear CG method by Fletcher and Reeves is however that we do not choose $z_k \coloneqq x_k + \alpha_k d_k$ as the next iterate. Instead, in Step 4, we employ a hyperplane projection technique Solodov and Svaiter introduced for their inexact Newton method to achieve global convergence [71]. They noticed that z_k and any solution $x^* \in Z$ are separated by the hyperplane

$$H = \{x \in \mathbb{R}^n \mid \langle F(z_k), x - z_k \rangle = 0\},$$

as

$$\langle F(z_k), x^* - z_k \rangle = -\langle F(x^*) - F(z_k), x^* - z_k \rangle \leq 0 \tag{4.4}$$

by the monotonicity of F and

$$\langle F(z_k), x_k - z_k \rangle = -\alpha_k \langle F(z_k), d_k \rangle > 0 \tag{4.5}$$

by any of the two line search conditions. Often, CG methods do not employ the extra parameter γ, that is the case $\gamma = 1$ where x_k is projected directly onto H. The only methods which allow a varying γ we are aware of are [2, 31], which happen to be the two methods we analyse in one of the following sections. We will see, why we need $0 < \gamma < 2$ in Lemma 4.4. After the projection to the hyperplane, we project onto Ω to enforce the constraint $x \in \Omega$. In particular, (4.4) also implies

$$\langle F(z_k), x_k - z_k \rangle = \langle F(z_k), x_k - x^* \rangle + \langle F(z_k), x^* - z_k \rangle \leq \langle F(z_k), x_k - x^* \rangle. \tag{4.6}$$

This leads us to the following lemma.

Lemma 4.3. *(see [77, Lemma 2.2]) Let $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm 4.1 and $x^* \in Z$. Then, $-F(z_k)$ is descent direction of $\|x - x^*\|^2$ at x_k.*

Proof. Let $h(x) = \|x - x^*\|^2$. Then, by (4.6) and (4.5),

$$\langle -F(z_k), \nabla h(x_k) \rangle = -2\langle F(z_k), x_k - x^* \rangle \leq -2\langle F(z_k), x_k - z_k \rangle < 0$$

holds and thus $-F(z_k)$ is a descent direction for $\|x - x^*\|^2$. □

This result guarantees that projecting x_k along $F(z_k)$ brings the iterate indeed closer to the solution set Z (or at least, not further away). Furthermore, we can actually guarantee that $\|x_k - x^*\|$ is monotonically decreasing and thus convergent. The following theorem for the case $\gamma = 1$ can be found in [77, Lemma 2.3]. The variant with a varying γ, we have adapted from [31, Lemma 3.2].

Lemma 4.4. *Let $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm 4.1 and let $x^* \in Z$. Then*

$$\begin{aligned}\|x_{k+1} - x^*\|^2 &\leq \|x_k - x^*\|^2 - \gamma(2-\gamma)\frac{\langle F(z_k), x_k - z_k\rangle^2}{\|F(z_k)\|^2} \\ &\leq \|x_k - x^*\|^2.\end{aligned}$$

In particular, $\{x_k\}$ is then bounded, as $\|x_k - x^*\| \leq \|x_0 - x^*\|$ for all k, and $\{\|x_k - x^*\|\}$ is convergent.

Proof. By using the fact that projections are Lipschitz continuous with Lipschitz-constant $L = 1$, then (4.6), and finally $0 < \gamma < 2$, we get

$$\begin{aligned}\|x_{k+1} - x^*\|^2 &= \|P_\Omega[x_k - \gamma\mu_k F(z_k)] - P_\Omega[x^*]\|^2 \\ &\leq \|x_k - \gamma\mu_k F(z_k) - x^*\|^2 \\ &= \|x_k - x^*\|^2 - 2\gamma\mu_k\langle F(z_k), x_k - x^*\rangle + \gamma^2\mu_k^2\|F(z_k)\|^2 \\ &\leq \|x_k - x^*\|^2 - 2\gamma\mu_k\langle F(z_k), x_k - z_k\rangle + \gamma^2\mu_k^2\|F(z_k)\|^2 \\ &= \|x_k - x^*\|^2 - \gamma(2-\gamma)\frac{\langle F(z_k), x_k - z_k\rangle^2}{\|F(z_k)\|^2} \\ &\leq \|x_k - x^*\|^2,\end{aligned}$$

which is the desired inequality. Furthermore, $\{x_k\}$ is bounded and the sequence $\{\|x_k - x^*\|\}$ is convergent, as it is monotone and bounded. □

Thus, using $x_k - \gamma\mu_k F(z_k)$ (or the projection thereof) as the next iterate x_{k+1} brings it indeed closer to the solution set Z for any $\gamma \in (0, 2)$. The next lemma shows how the iterates $\{z_k\}$ behave relative to $\{x_k\}$. It is also the first result where we need to involve the line search conditions directly. For both line search conditions considered in this thesis, we thus need separate proofs. Our proof for line search (L1) follows [2, Lemma 2.6], although we eliminate the need of F being Lipschitz to get that $\{F_k\}$ and $\{F(z_k)\}$ are bounded. For line search (L2), we adapted part of the proof of [31, Lemma 3.2].

Lemma 4.5. *Let $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm 4.1. Then, we have*

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

which is equivalent to

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

Proof. Let $x^* \in Z$. We will start with the proof for line search (L1). First, note that $\{F_k\}$ is bounded due to the fact that F is continuous and that $\{x_k\}$ is bounded. That is, there exists a $C > 0$, such that

$$\|F_k\| \leq C$$

for all k. On the one hand, we now get

$$\langle F(z_k), x_k - z_k\rangle = -\alpha_k \langle F(z_k), d_k\rangle \geq \kappa \alpha_k^2 \|d_k\|^2 = \kappa \|x_k - z_k\|^2.$$

On the other hand, by the monotonicity of F and Cauchy-Schwarz, we obtain

$$\langle F(z_k), x_k - z_k\rangle \leq \langle F(x_k), x_k - z_k\rangle \leq \|F(x_k)\|\|x_k - z_k\|.$$

Together, these yield

$$\kappa \|x_k - z_k\|^2 \leq \|F(x_k)\|\|x_k - z_k\|,$$

which, by the boundedness of $\{F_k\}$, implies that

$$\|x_k - z_k\| \leq \frac{C}{\kappa}.$$

With this and the boundedness of $\{x_k\}$, we get that $\{z_k\}$ is also bounded due to

$$\begin{aligned}\|z_k - x^*\| &\leq \|z_k - x_k\| + \|x_k - x^*\| \\ &\leq \frac{C}{\kappa} + \|x_0 - x^*\|.\end{aligned}$$

Thus, $\{F(z_k)\}$ is also bounded by the continuity of F, i.e., there exists a $D > 0$, auch that

$$\|F(z_k)\| \leq D$$

for all k. The line search condition (L1) and Lemma 4.4 finally yield

$$\begin{aligned}\kappa^2 \alpha_k^4 \|d_k\|^4 &\leq \alpha_k^2 \langle F(z_k), d_k\rangle^2 \\ &= \langle F(z_k), z_k - x_k\rangle^2 \\ &\leq \frac{\|F(z_k)\|^2}{\gamma(2-\gamma)}(\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2) \\ &\leq \frac{D^2}{\gamma(2-\gamma)}(\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2),\end{aligned}$$

where the right hand side tends to zero, as $\{\|x_k - x^*\|\}$ is convergent by Lemma 4.4. This implies the assertion.

For line search (L2), the proof is slightly simpler. By Lemma 4.4 and (L2), we obtain the inequality

$$\begin{aligned}\|x_{k+1} - x^*\|^2 &\leq \|x_k - x^*\|^2 - \gamma(2-\gamma)\frac{\langle F(z_k), x_k - z_k\rangle^2}{\|F(z_k)\|^2} \\ &\leq \|x_k - x^*\|^2 - \gamma(2-\gamma)\kappa^2 \alpha_k^4 \|d_k\|^4 \\ &= \|x_k - x^*\|^2 - \gamma(2-\gamma)\kappa^2 \|x_k - z_k\|^4,\end{aligned}$$

and thus

$$\gamma(2-\gamma)\kappa^2\|x_k - z_k\|^4 \leq \|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2.$$

Again, the right hand side converges to zero, yielding the assertion

$$\lim_{k\to\infty} \|x_k - z_k\| = 0. \qquad \square$$

Finally, before we go to the next section, we want to briefly mention, that we have observed that some authors only show that

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

This however is enough for the convergence of $\{x_k\}$ to a solution.

Lemma 4.6. *Let $\{x_k\}$ be generated by Algorithm 4.1 such that*

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

Then, $\{x_k\}$ converges to a solution $\bar{x} \in Z$ and

$$\lim_{k\to\infty} F(x_k) = 0.$$

Proof. Let $\{\|F(x_{k_l})\|\}$ be the subsequence converging towards 0. Then, $\{x_{k_l}\}$ contains an accumulation point $\bar{x}$ by Bolzano-Weierstrass, as $\{x_k\}$ is bounded by Lemma 4.4. By the continuity of F, this then implies that $\bar{x} \in Z$. Using Lemma 4.4 again however, $\{\|x_k - x^*\|\}$ converges towards its infimum. As the subsequence $\{\|x_{k_l} - x^*\|\}$ already converges to zero, we also have that $\{\|x_k - x^*\|\}$ converges to zero. Thus, $\{x_k\}$ converges towards the solution $\bar{x}$. The last assertion follows from the continuity of F. $\square$

4.2 Additional Conditions and their Impact on Convergence

While the last section shows various nice properties of Conjugate Gradient methods using the general framework given in Algorithm 4.1, these do not suffice to guarantee actual convergence of the algorithm to a solution. It is difficult to provide general convergence results for all CG methods. Thus, search directions d_k have to be carefully crafted to ensure convergence. There is however one condition that is very helpful and commonly used, that is the *sufficient descent condition*. It often plays a crucial role in proving convergence.

The search directions $\{d_k\}$ are said to satisfy the sufficient descent condition, if there exists a $c > 0$, such that

$$F_k^\top d_k \leq -c\|F_k\|^2 \tag{4.7}$$

for all k. Using Cauchy-Schwarz, we also obtain

$$c\|F_k\|^2 \leq -F_k^\top d_k \leq \|F_k\|\|d_k\|,$$

and thus

$$c\|F_k\| \leq \|d_k\|. \tag{4.8}$$

Clearly, this condition is stronger than the regular descent condition (4.3). Authors try to construct search directions, that satisfy this property. Another very common additional assumption on F is the Lipschitz continuity, that is, there exists an $L > 0$, such that

$$\|F(x) - F(y)\| \leq L\|x - y\| \tag{4.9}$$

for all $x, y \in \mathbb{R}^n$.

An example for an application of the sufficient descent condition is the following theorem. Note that this is again dependent on the line search condition.

Theorem 4.7. *(cf. [2, Lemma 2.5], [31, Lemma 3.3]) Let $\{x_k\}$ and $\{\alpha_k\}$ be generated by Algorithm 4.1. Given that the sufficient descent condition* (4.7) *holds and that F is Lipschitz continuous, we have the inequality*

$$\alpha_k \geq \min\left\{\xi, \frac{c\rho\|F_k\|^2}{(L + M\kappa)\|d_k\|^2}\right\},$$

with a constant $M > 0$.

Proof. For $\alpha_k = \xi$, the assertion is clear. Otherwise, we have $\alpha_k < \xi$. First, we argue that $\{F(x_k + \rho^{-1}\alpha_k d_k)\}$ is bounded. For this, consider the inequality

$$\|x_k + \rho^{-1}\alpha_k d_k - x^*\| \leq \|x_k - x^*\| + \rho^{-1}\alpha_k\|d_k\| = \|x_k - x^*\| + \rho^{-1}\|x_k - z_k\|.$$

As the right hand side converges, the sequence $\{x_k + \rho^{-1}\alpha_k d_k\}$ is bounded. Thus, by the continuity of F, the sequence $\{F(x_k + \rho^{-1}\alpha_k d_k)\}$ is bounded.

Since $\alpha_k < \xi$, we know that $\rho^{-1}\alpha_k$ does not fulfill the line search condition, i.e.,

$$-F(x_k + \rho^{-1}\alpha_k d_k)^\top d_k < \kappa\rho^{-1}\alpha_k\|d_k\|^2$$

for line search (L1) and

$$-F(x_k + \rho^{-1}\alpha_k d_k)^\top d_k < \kappa\rho^{-1}\alpha_k\|F(x_k + \rho^{-1}\alpha_k d_k)\|\|d_k\|^2$$

for line search (L2). Hence, we have that

$$-F(x_k + \rho^{-1}\alpha_k d_k)^\top d_k < \kappa\rho^{-1}\alpha_k M\|d_k\|^2, \tag{4.10}$$

where we set $M = 1$ for line search (L1) and M as the upper bound of $\{\|F(x_k+\rho^{-1}\alpha_k d_k)\|\}$ for line search (L2). The sufficient descent condition (4.7), (4.10) and the Lipschitz continuity of F now finally yield

$$\begin{aligned} c\|F(x_k)\|^2 &\leq -F(x_k)^\top d_k \\ &= (F(x_k + \rho^{-1}\alpha_k d_k)^\top d_k - F(x_k)^\top d_k) - F(x_k + \rho^{-1}\alpha_k d_k)^\top d_k \\ &< \|F(x_k + \rho^{-1}\alpha_k d_k) - F(x_k)\|\|d_k\| - M\kappa\rho^{-1}\alpha_k\|d_k\|^2 \\ &\leq (L + M\kappa)\rho^{-1}\alpha_k\|d_k\|^2. \end{aligned}$$

This implies the assertion. □

While this theorem on its own is still not enough to prove convergence, having a lower bound on α_k is often a helpful property. For example, it immediately provides convergence, as soon as the search directions d_k are bounded.

Theorem 4.8. *Let $\{x_k\}$ be generated by Algorithm 4.1, d_k satisfy the sufficient descent condition* (4.7) *and F be Lipschitz. If the search direction are bounded, then $\{x_k\}$ converges to a solution.*

Proof. By Lemma 4.6, it suffices to show that $\liminf_{k\to\infty} \|F(x_k)\| = 0$. Assume that this assertion is false, which implies the existence of an $\varepsilon > 0$ such that

$$\|F(x_k)\| \geq \varepsilon \quad \forall k \in \mathbb{N}_0.$$

Through (4.8), we thus get

$$\|d_k\| \geq c\varepsilon \quad \forall k \in \mathbb{N}_0,$$

and therefore

$$\lim_{k\to\infty} \alpha_k = 0$$

by Lemma 4.5. However, as $\|F(x_k)\|$ is now bounded from below by ε and $\|d_k\|$ bounded form above, the α_k are bounded away from zero by Theorem 4.7. This poses a contradiction and thus completes the proof. □

Another additional property we have encountered, e.g., in [2] discussed in the next section, is the converse of (4.8). That is, there exists a $\vartheta >$ such that

$$\|d_k\| \leq \vartheta \|F_k\| \tag{4.11}$$

for all k. As $\{F_k\}$ is bounded, it is clear that this property immediately implies convergence of $\{x_k\}$ to a solution by Theorem 4.8. Additionally, it implies by Theorem 4.7 that α_k is bounded away from zero by a constant. This is very useful, as it guarantees the termination of the line search after a fixed amount of steps.

In the case that convergence of $\{x_k\}$ to a solution $\bar{x}$ is already known, (4.11) also allows us to prove linear convergence in the case that F is Lipschitz and satisfies the error bound condition

$$\operatorname{dist}[x, Z] \leq \ell \|F(x)\| \tag{4.12}$$

around a neighborhood of the solution $\bar{x}$ with an $\ell > 0$. For this proof, we however need the line search strategy (L1). The proof is taken from [2, Theorem 2.10] but slightly adapted to work with arbitrary values of ξ and κ. We also spell out some arguments.

Theorem 4.9. *Let $\{x_k\}$ be generated by Algorithm 4.1 using the line search* (L1)*. Assume that $\{x_k\}$ converges to a solution $\bar{x}$ and that* (4.7)*,* (4.9)*,* (4.11) *and* (4.12) *hold. Then $\{\operatorname{dist}[x_k, Z]\}$ converges Q-linear to zero and $\{x_k\}$ converges R-linear to $\bar{x}$.*

Proof. For $k \in \mathbb{N}_0$, let $\bar{x}_k \in Z$ such that $\operatorname{dist}[x_k, Z] = \|x_k - \bar{x}_k\|$. Now, with (4.9) and (4.11), we get

$$\begin{aligned}
\|F(z_k)\| &= \|F(z_k) - F(\bar{x}_k)\| \\
&\leq L\|z_k - \bar{x}_k\| \\
&\leq L(\|z_k - x_k\| + \|x_k - \bar{x}_k\|) \\
&= L(\alpha_k\|d_k\| + \|x_k - \bar{x}_k\|) \\
&\leq L(\xi\|d_k\| + \|x_k - \bar{x}_k\|) \\
&\leq L(\xi\vartheta\|F(x_k) - F(\bar{x}_k)\| + \|x_k - \bar{x}_k\|) \\
&\leq L(L\xi\vartheta + 1)\|x_k - \bar{x}_k\| \\
&\leq L(L\xi\vartheta + 1)\operatorname{dist}[x_k, Z].
\end{aligned} \tag{4.13}$$

Let k be sufficiently large, such that the iterates x_k lie within the neighborhood of $\bar{x}$ where the error bound condition (4.12) holds. If we apply Lemma 4.4, (L1), (4.7), (4.12) and then finally (4.13) we get

$$\begin{aligned}
\operatorname{dist}[x_{k+1}, Z]^2 &= \|x_{k+1} - \bar{x}_k\|^2 \\
&\leq \|x_k - \bar{x}_k\|^2 - \gamma(2-\gamma)\frac{\langle F(z_k), x_k - z_k\rangle^2}{\|F(z_k)\|^2} \\
&\leq \|x_k - \bar{x}_k\|^2 - \gamma(2-\gamma)\frac{\kappa^2\alpha_k^4\|d_k\|^4}{\|F(z_k)\|^2} \\
&\leq \|x_k - \bar{x}_k\|^2 - \gamma(2-\gamma)\frac{\kappa^2\alpha_k^4 c^4\|F_k\|^4}{\|F(z_k)\|^2} \\
&\leq \|x_k - \bar{x}_k\|^2 - \gamma(2-\gamma)\frac{\kappa^2\alpha_k^4 c^4\ell^{-4}\operatorname{dist}[x_k, Z]^4}{L^2(L\xi\vartheta+1)^2\operatorname{dist}[x_k, Z]^2} \\
&= \left(1 - \gamma(2-\gamma)\frac{\kappa^2\alpha_k^4 c^4\ell^{-4}}{L^2(L\xi\vartheta+1)^2}\right)\operatorname{dist}[x_k, Z]^2.
\end{aligned}$$

Without loss of generality, we can increase L and ℓ such that

$$L > \kappa\xi^2 c^2 \quad \text{and} \quad \ell > 1 \tag{4.14}$$

hold. With that, we have $\gamma(2-\gamma) \in (0,1)$, $\kappa^2\alpha_k^4 c^4/L \in (0,1)$, $\ell^{-4} \in (0,1)$ and $(L\xi\vartheta+1)^2 > 1$. Additionally, by Theorem 4.7, we have that α_k is bounded from below, i.e., $\alpha_k \geq \alpha > 0$. Thus, it holds that

$$\operatorname{dist}[x_{k+1}, Z]^2 \leq \left(1 - \gamma(2-\gamma)\frac{\kappa^2\alpha^4 c^4\ell^{-4}}{L^2(L\xi\vartheta+1)^2}\right)\operatorname{dist}[x_k, Z]^2,$$

where

$$1 - \gamma(2-\gamma)\frac{\kappa^2\alpha^4 c^4\ell^{-4}}{L^2(L\xi\vartheta+1)^2} \in (0,1).$$

Thus, we have proven the Q-linear convergence of $\{\operatorname{dist}[x_k, Z]\}$ and we further know that $\{x_k\}$ converges R-linearly to $\bar{x}$. □

4.3 Two Conjugate Gradient Methods

In this section, we will look at two recent conjugate gradient algorithm, the NDK method by Ahmed et al. [2] and the algorithm by Gao and He [31]. We were interested in these two methods, as their authors claim linear convergence under an error bound condition, that we already discussed in Chapter 3.

We also take a look at NDK in particular, as they employ a technique inspired by Perry [60] to find an optimal choice of a variable parameter in their search direction. As we will see in section 5.3, this method perform very well.

Also note that both of these methods are actually three-term methods, that is they include a third vector term in their search direction.

4.3.1 A Symmetric Dai-Kou Based Method

The NDK method introduced by Ahmed et al. [2] is a method based on the Conjugate Gradient method by Dai and Kou [16]. As already stated in the introduction, this method was used to solve minimization problems and used a CG parameter of the form

$$\beta_{k+1}^{\mathrm{DK}} = \frac{\nabla f_{k+1}^{\top} y_k}{y_k \top d_k} - \left(\tau_k + \frac{\|y_k\|^2}{s_k^{\top} y_k} - \frac{s_k^{\top} y_k}{\|s_k\|^2}\right) \frac{\nabla f_{k+1}^{\top} s_k}{y_k \top d_k},$$

with a scalar parameter τ_k and the common abbreviations $y_k \coloneqq \nabla f_{k+1} - \nabla f_k$ and $s_k \coloneqq x_{k+1} - x_k$. They have previously also proposed a Dai-Kou scheme in [79] that they now improve upon. Their new CG parameter is the Dai-Kou parameter

$$\beta_k^{\mathrm{NDK}} = \frac{F_{k+1}^{\top} \bar{y}_k}{d_k^{\top} \bar{y}_k} - \left(\tau_k + \frac{\|\bar{y}_k\|^2}{\bar{s}_k^{\top} \bar{y}_k} - \frac{\bar{s}_k^{\top} \bar{y}_k}{\|\bar{s}_k\|^2}\right) \frac{F_{k+1}^{\top} \bar{s}_k}{d_k^{\top} \bar{y}_k}$$

with a modified $\bar{y}_k$. The vectors $\bar{y}_k$ and $\bar{s}_k$ are given by

$$\bar{y}_k \coloneqq y_k + \varrho_k \bar{s}_k + G\|F_k\|^r \bar{s}_k, \; y_k \coloneqq F(z_k) - F(x_k), \; \bar{s}_k \coloneqq z_k - x_k,$$

with

$$\varrho_k \coloneqq \max\left\{-\frac{\bar{s}_k^{\top} \bar{y}_k}{\|\bar{s}_k\|^2}, 0\right\}.$$

Their search direction is then defined by

$$d_{k+1} = -F_{k+1} + \beta_k^{\mathrm{NDK}} d_k + \frac{F_{k+1}^{\top} \bar{s}_k}{\bar{s}_k^{\top} \bar{y}_k} \bar{y}_k,$$

where they also added a third third, making this a three-term method. As a line search strategy, they use (L1) from Algorithm 4.1

To find the optimal τ_k, the authors use an approach inspired by Perry [60] that is sometimes used by the authors of CG methods to motivate a choice of some variable parameter. They use the fact that d_{k+1} may be written as

$$d_{k+1} = -Q_{k+1} F_{k+1},$$

with a matrix $Q_{k+1} \in \mathbb{R}^{n\times n}$. This poses similarities to Quasi-Newton methods as the matrix Q_{k+1} can be seen as an approximation of the inverse Jacobian. For other methods that employ this technique, see e.g., [21, 67, 78, 79].

In this case, the matrix Q_{k+1} is given by

$$Q_{k+1} = I - \frac{\bar{s}_k \bar{y}_k^\top}{\bar{s}_k^\top \bar{y}_k} - \frac{\bar{y}_k \bar{s}_k^\top}{\bar{s}_k^\top \bar{y}_k} + \tau_k \frac{\bar{s}_k \bar{s}_k^\top}{\bar{s}_k^\top \bar{y}_k} + \frac{\|\bar{y}_k\|^2 \bar{s}_k \bar{s}_k^\top}{(\bar{s}_k^\top \bar{y}_k)^2} - \frac{\bar{s}_k \bar{s}_k^\top}{\|\bar{s}_k\|^2},$$

using the fact that $\bar{s}_k = \alpha_k d_k$. A special thing to note about the matrix for this particular method is that it is symmetric. For other methods, this is usually not the case and those authors then define

$$\tilde{Q}_{k+1} \coloneqq \frac{Q_{k+1} + Q_{k+1}^\top}{2}$$

to obtain a symmetric matrix. For this method however, this is not necessary. The matrix is furthermore a rank-two update of the identity matrix, namely

$$Q_{k+1} = I + u_1 u_2^\top + u_3 u_4^\top,$$

with

$$u_1 = -\bar{y}_k, \qquad u_2 = \frac{\bar{s}_k}{\bar{s}_k^\top \bar{y}_k}, \qquad u_3 = -\frac{\bar{s}_k}{(\bar{s}_k^\top \bar{y}_k)^2}$$

and

$$u_4 = \frac{(\bar{s}_k^\top \bar{y}_k \|\bar{s}_k\|^2 \bar{y}_k - \tau_k \bar{s}_k^\top \bar{y}_k \|\bar{s}_k\|^2 \bar{s}_k - \|\bar{s}_k\|^2 \|\bar{y}_k\|^2 \bar{s}_k + (\bar{s}_k^\top \bar{y}_k)^2 \bar{s}_k)^\top}{\|\bar{s}_k\|^2}.$$

After this follows an analysis of the eigenvalues and arguments for a suitable parameter. The common approach to find the eigenvalues goes as follows.

There exists a set of orthonormal vectors $\zeta_k^1, \dots, \zeta_k^{n-2}$ such that

$$\langle \bar{s}_k, \zeta_k^i \rangle = \langle \bar{y}_k, \zeta_k^i \rangle = 0.$$

Therefore

$$Q_{k+1} \zeta_k^i = \zeta_k^i,$$

and Q_{k+1} has the eigenvalue 1 with multiplicity at least $n-2$. Let η_k^+ and η_k^- be the remaining eigenvalues with $\eta_k^- \leq \eta_k^+$. Since the trace is the sum of the eigenvalues, we get

$$n - 2 + \eta_k^- + \eta_k^+ = \operatorname{tr}(Q_{k+1}) = n - 2 + \tau_k \frac{\|\bar{s}_k\|^2}{\bar{s}_k^\top \bar{y}_k} + \frac{\|\bar{y}_k\|^2 \|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2} - 1,$$

which yields

$$\eta_k^+ + \eta_k^- = \tau_k \frac{\|\bar{s}_k\|^2}{\bar{s}_k^\top \bar{y}_k} + \frac{\|\bar{y}_k\|^2 \|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2} - 1. \tag{4.15}$$

In order to be able to solve for η_k^+ and η_k^-, we are going to use the determinant of Q_{k+1} to obtain a second equation. To calculate the determinant, we use the following lemma.

Lemma 4.10. *(cf. [72, eq. 1.2.70]) For $u_1, u_2, u_3, u_4 \in \mathbb{R}^n$, it holds that*

$$\det(I + u_1 u_2^\top + u_3 u_4^\top) = (1 + u_1^\top u_2)(1 + u_3^\top u_4) - (u_1^\top u_4)(u_2^\top u_3).$$

Proof. Using Sylvester's determinant theorem [63, eq. (B.1.15)]

$$\det(I_n + AB) = \det(I_m + BA)$$

for all $A^\top, B \in \mathbb{R}^{m\times n}$, we get

$$\begin{aligned}
\det(I + u_1u_2^\top + u_3u_4^\top) &= \det\left(I_n + \begin{pmatrix} u_1 & u_3 \end{pmatrix}\begin{pmatrix} u_2^\top \\ u_4^\top \end{pmatrix}\right) \\
&= \det\left(I_2 + \begin{pmatrix} u_2^\top \\ u_4^\top \end{pmatrix}\begin{pmatrix} u_1 & u_3 \end{pmatrix}\right) \\
&= \det\left(I_2 + \begin{pmatrix} u_1^\top u_2 & u_2^\top u_3 \\ u_1^\top u_4 & u_3^\top u_4 \end{pmatrix}\right) \\
&= (1 + u_1^\top u_2)(1 + u_3^\top u_4) - (u_1^\top u_4)(u_2^\top u_3).
\end{aligned}$$

□

Note that we constructed a different proof for this lemma, as the original proof of [72] makes invertibility assumptions. This proof is thus more general.

Using this lemma with

$$u_1^\top u_2 = -1, \qquad u_1^\top u_4 = \tau_k(\bar{s}_k^\top \bar{y}_k)^2 - \frac{(\bar{s}_k^\top \bar{y}_k)^3}{\|\bar{s}_k\|^2}, \qquad u_2^\top u_3 = -\frac{\|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^3},$$

we then obtain

$$\det(Q_{k+1}) = \tau_k \frac{\|\bar{s}_k\|^2}{\bar{s}_k^\top \bar{y}_k} - 1,$$

and as the determinant is also the product of the eigenvalues, we thus have

$$\eta_k^+ \eta_k^- = \tau_k \frac{\|\bar{s}_k\|^2}{\bar{s}_k^\top \bar{y}_k} - 1. \tag{4.16}$$

To guarantee that d_{k+1} does indeed satisfy the descent condition (4.3), Q_{k+1} must be positive definite, so we require

$$\eta_k^+, \eta_k^- > 0,$$

which is equivalent to

$$\tau_k > \frac{\bar{s}_k^\top \bar{y}_k}{\|\bar{s}_k\|^2}. \tag{4.17}$$

By setting

$$\eta_k^\pm = \frac{1}{2}(\bar{\eta} \pm \chi),$$

we directly get from (4.15) that

$$\bar{\eta} = \tau_k \frac{\|\bar{s}_k\|^2}{\bar{s}_k^\top \bar{y}_k} + \frac{\|\bar{y}_k\|^2\|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2} - 1.$$

Then, inserting into (4.16) yields

$$\frac{1}{4}(\bar{\eta}^2 - \chi^2) = \tau_k \frac{\|\bar{s}_k\|^2}{\bar{s}_k^\top \bar{y}_k} - 1 = \bar{\eta} - \frac{\|\bar{y}_k\|^2\|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2},$$

from which we can conclude

$$\begin{aligned}\chi^2 &= \bar{\eta}^2 - 4\bar{\eta} + 4\frac{\|\bar{y}_k\|^2\|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2} \\ &= (\bar{\eta}-2)^2 + 4\frac{\|\bar{y}_k\|^2\|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2} - 4 \\ &= \left(\tau_k \frac{\|\bar{s}_k\|^2}{\bar{s}_k^\top \bar{y}_k} + \frac{\|\bar{y}_k\|^2\|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2} - 3\right)^2 + 4\frac{\|\bar{y}_k\|^2\|\bar{s}_k\|^2}{(\bar{s}_k^\top \bar{y}_k)^2} - 4.\end{aligned}$$

Now that we have closed formulas for $\eta_k^\pm$, we need to find a suitable τ_k. First, we set

$$\tau_k > \frac{\|\bar{y}_k\|^2}{\bar{s}_k^\top \bar{y}_k}, \tag{4.18}$$

as (4.17) is then automatically satisfied by Cauchy-Schwarz, as well as

$$\chi^2 \geq (\bar{\eta}-2)^2.$$

Through this, we get the upper bound

$$\eta_k^- = \frac{1}{2}(\bar{\eta} - \chi) \leq \frac{1}{2}(\bar{\eta} - |\bar{\eta} - 2|) \leq 1.$$

Likewise, it is also easy to see that $\bar{\eta} > 1$ and therefore

$$\eta_k^+ = \bar{\eta} - \eta_k^- > 2 \geq 1.$$

Together, we have that

$$0 < \eta_k^- \leq 1 < \eta_k^+,$$

and therefore these two eigenvalues are the smallest and largest eigenvalues respectively. Additionally, Q_{k+1} is positive definite, allowing us to show the descent condition

$$d_{k+1}^\top F_{k+1} = -F_{k+1}^\top Q_{k+1} F_{k+1} \leq -\eta_k^- \|F_{k+1}\|^2 < 0. \tag{4.19}$$

Finally, as we still have much freedom on the choice of τ_k, we have the opportunity to tune it to improve numerical stability and therefore performance. Thus, we examine the condition number of Q_{k+1} given as the quotient of the largest and smallest eigenvalue

$$\operatorname{cond}(Q_{k+1}) = \frac{\eta_k^+}{\eta_k^-}.$$

Hence, we choose τ_k to minimize the condition number and thus achieve high numerical stability. As the quotient becomes smaller when the difference χ between η_k^+ and η_k^- becomes smaller, we thus minimize χ. It is easy to see that this is the case when $\bar{\eta} = 2$. Solving for τ_k yields

$$\tau_k^* = \arg\min(\chi) = \frac{3\bar{s}_k^\top \bar{y}_k}{\|\bar{s}_k\|^2} - \frac{\|\bar{y}_k\|^2}{\bar{s}_k^\top \bar{y}_k}.$$

As we also still need to ensure (4.18) holds, the authors suggested these two possible values for τ_k

$$\tau_k^1 = \max\left\{\tau_k^*, q_1 \frac{\|\bar{y}_k\|^2}{\bar{s}_k^\top \bar{y}_k}\right\},$$

and

$$\tau_k^2 = \max\left\{\tau_k^*, q_2 + \frac{\|\bar{y}_k\|^2}{\bar{s}_k^\top \bar{y}_k}\right\},$$

with $q_1 > 1$, $q_2 > 0$. These are the two possible choices of parameters used in the author's method.

Note that the authors at some point, without further explanation, replace the k-dependent eigenvalue η_k^- by a constant $\eta > 0$ [2, page 11, last line]. This is important, as they use boundedness away from zero in their convergence proof [2, eq. (2.44)]. This might not be the case for $\{\eta_k^-\}$. We do not understand how one would prove the existence of one such bound, and thus we believe

$$\eta_k^- \geq \eta, \quad \forall k \in \mathbb{N}_0$$

for some $\eta > 0$ should be appear as an *assumption* in [2, Theorem 2.7]. As we see in our numerical analysis that the theoretical results resulting from this assumption hold, we conclude that this assumption is sensible.

From this assumption and (4.19), we can conclude that the sufficient descent condition (4.7) holds for $c = \eta$, i.e.,

$$F_k^\top d_k \leq -\eta \|F_k\|^2 \tag{4.20}$$

for all k. With this and a Lipschitz assumption, the authors manage to prove (4.11), that is, there exists a $\vartheta > 0$, such that

$$\|d_k\| \leq \vartheta \|F_k\|.$$

Thus we know from Section 4.2, that $\{x_k\}$ converges indeed to a solution.

With this, the author further prove the Q-linear convergence of $\{\text{dist}[x_k, z]\}$ to zero and R-linear convergence of $\{x_k\}$ to $\bar{x}$. Have have included this proof in Theorem 4.9.

4.3.2 An Efficient Three Term CG Method by Gao and He

In this section, we briefly discuss the three-term CG algorithm by Gao and He [31]. As we have already provided most of the theorems in previous sections, we only need to include the parts specific to this algorithm. The search direction of this algorithm is given by

$$d_k = -F_k + \beta_k d_{k-1} + \theta_k F_{k-1},$$

where

$$\beta_k = -\frac{F_k^\top F_{k-1}}{F_{k-1}^\top d_{k-1}}, \quad \theta_k = \frac{F_k^\top d_{k-1}}{F_{k-1}^\top d_{k-1}}.$$

This search direction is inspired by Zhang et al. [84], who used the search direction

$$d_k = -F_k + \frac{F_k^\top y_{k-1}}{d_{k-1}^\top y_{k-1}} d_{k-1} - \frac{F_k^\top d_{k-1}}{d_{k-1}^\top y_{k-1}} y_{k-1},$$

with $y_{k-1} \coloneqq F_k - F_{k-1}$. The new method is interesting in that its parameters only consist of a single fraction with each containing two inner products, making calculation of the search direction slightly more efficient than other methods. As a line search strategy, this method uses (L2).

With the definition of the search direction, we can directly calculate

$$F_k^\top d_k = -\|F_k\|^2,$$

which is the sufficient descent condition (4.7) with equality instead of an inequality for $c = 1$. This condition also yields

$$\|F_k\| \leq \|d_k\|.$$

After providing this inequality, the authors present a calculation resulting in the inequality

$$\|d_k\| \leq 3\|F_k\|. \tag{4.21}$$

As we know from Section 4.2, this directly implies the convergence of the iterates to a solution. However, even after thorough review of the calculation, we do not see why the inequality (4.21) holds. The authors only use (4.21) to argue that the search directions d_k are bounded, which implies convergence by Theorem 4.8 when F is Lipschitz. As we however also failed to prove the boundedness of d_k without use of (4.21), we propose to make it an assumption that d_k is bounded. As we see convergence of the algorithm in the numerics, we belief this is a sensible assumption.

An additional concern we have is with [31, Theorem 4.1]. In addition to the Lipschitz assumption, they assume that the error bound condition

$$\operatorname{dist}[x, Z] \leq \ell \|F(x)\|$$

holds for an $\ell > 1$ around a neighborhood of the solution $\bar{x}$. The theorem then states that $\{\operatorname{dist}[x_k, Z]\}$ converges Q-linearly, while $\{x_k\}$ converges R-linearly. They first show the inequality

$$\operatorname{dist}[x_{k+1}, Z]^2 \leq (1 - \gamma(2-\gamma)s\ell^{-2}\alpha_k^2) \operatorname{dist}[x_k, Z]^2 \tag{4.22}$$

with

$$s = \frac{\langle F(z_k), x_k - z_k\rangle^2}{\|F(z_k)\|^2 \|x_k - z_k\|^2}. \tag{4.23}$$

After this, they argue that the factor $(1 - \gamma(2-\gamma)s\ell^{-2}\alpha_k^2)$ lies within $(0, 1)$ and conclude linear convergence. However, they do not provide an argument as to why this factor does not approach one, which would mean that $\{\operatorname{dist}[x_k, Z]\}$ is, in fact, not linearly convergent. While α_k is bounded by Theorem 4.7 if we assume (4.21), we do not see how one can rule out that s in (4.23) approaches zero. If that were the case, the factor in (4.22) would approach one, disproving linear convergence. After multiple attempts, we did not succeed in finding an alternative proof. Thus, we belief that additional assumptions or changes to the algorithm would be necessary to prove this theorem.

5

Numerical Experiments

All tests were implemented in Python using the NumPy library and performed on an Intel™ i5–1240P with 16 GB of RAM. The Linear Programs were solved using the CVXPY library with their SciPy backend. The code is available at [75].

In all of the tests, we consider systems of equations with the same number of equations as variables, that is $F\colon \mathbb{R}^n \to \mathbb{R}^m$ with $n = m$. The set of monotone problems on which we test are the following.

Problem 1. (obtained from [21])
$F_i(x) = 2c(x_i - 1) + 4(t - 0.25)x_i, \quad i = 1, 2, \dots n,$
where $t = \sum_{i=1}^{n} x_i^2, \quad c = 10^{-5}, \quad \Omega = \mathbb{R}^n_+.$

Problem 2. (obtained from [31])
$F_1(x) = x_1 - \exp(\cos \frac{x_1 + x_2}{2}),$
$F_i(x) = x_i - \exp(\cos \frac{x_{i-1} + x_i + x_{i+1}}{i}), \quad i = 2, 3, \dots, n-1$
$F_n(x) = x_n - \exp(\cos \frac{x_{n-1} + x_n}{n}),$
where $\Omega = \mathbb{R}^n_+.$

Problem 3. (obtained from [42])
$F_i(x) = \log(x_i + 1) - \frac{x_i}{n}, \quad i = 1, 2, \dots n,$
where $\Omega = \mathbb{R}^n_+.$

Problem 4. (obtained from [79])
$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1, 2, \dots, n,$
where $\Omega = \mathbb{R}^n_+.$

Problem 5. (obtained from [83])
$F_i(x) = \exp(x_i) - 1, \quad i = 1, 2, \dots, n,$
where $\Omega = \mathbb{R}^n_+.$

Problem 6. (obtained from [83])
$F_i(x) = x_i - \sin|x_i - 1|, \quad i = 1, 2, \dots, n,$
where $\Omega = \left\{ x \in \mathbb{R}^n \;\middle|\; \sum_{i=1}^{n} x_i \le n, \quad x \ge 0 \right\}.$

Note that e.g., Problem 3 is not actually monotonically increasing on its entire domain, but only when $\sum_{i=1}^{n} x_i \leq n$. As n is large enough in our tests however, this will not affect us.

Additionally, we also include the following non-monotone problems for tests of LP-Newton and SMLP-Newton.

Problem 7. (obtained from [53])
$F(x) = \left(x_2^2,\ x_2(1+x_1^2)\right)^\top$,
where $n = 2$ and $\Omega = \mathbb{R} \times \mathbb{R}_+$.

Problem 8. (obtained from [53])
$$F(x) = \Big(-x_1 - x_2 + 1 + x_3,\ -x_1^2 - x_2^2 + 1 + x_4,\\ -9x_1^2 - x_2^2 + 9 + x_5,\ -x_1^2 + x_2 + x_6,\ -x_2^2 + x_1 + x_7\Big)^\top,$$
where $n = 7$ and $\Omega = [-50, 50]^2 \times \mathbb{R}_+^5$.

Problem 9. (obtained from [24])
$$F(x) = \begin{pmatrix} x_1 x_2 - x_3 \\ x_1^2 + x_2 - 1 - x_4 \\ \min(x_1, x_3) \\ \min(x_2, x_4) \end{pmatrix},$$
where $n = 4$ and $\Omega = \mathbb{R}_+^4$.

Problem 10. (obtained from [24])
$$F(x) = \begin{pmatrix} x_4 + x_5 - x_6 - x_9 \\ x_4 + x_2 + x_3 - x_7 - x_9 \\ x_2 + x_3 - x_9 \\ x_1 + x_2 - x_8 \\ x_1 + x_{10} \\ -x_1 + x_{11} \\ 1 - x_2 + x_{12} \\ -x_5 + x_{13} \\ -x_1 - x_2 - x_3 + x_{14} \\ \min(x_5, x_{10}) \\ \min(x_6, x_{11}) \\ \min(x_7, x_{12}) \\ \min(x_8, x_{13}) \\ \min(x_9, x_{14}) \end{pmatrix},$$
where $n = 14$ and $\Omega = \mathbb{R}^4 \times \mathbb{R}_+^{10}$.

Problem 11. (obtained from [53])
$$F_i(x) = x_i - \frac{2}{n}\left(\sum_{j=1}^{n} x_j\right) - 1 + (x_i + 1)^2, \quad i = 1, \ldots, n,$$
where $\Omega = [-10, 10]^n$.

The first four of these problems do not allow a varying n, which will however not have a negative impact on our benchmarks. Also note that Facchinei et al. [24] originally obtained Problems 9 and 10 as KKT systems of complementary problems.

	LP-Newton			SMLP-Newton		
Problem	Iters	Time	$\|F(x^*)\|$	Iters	Time	$\|F(x^*)\|$
Problem 1	12	0.085	3.76×10^{-12}	16	0.120	6.38×10^{-12}
Problem 3	5	0.026	2.08×10^{-11}	1	0.005	0
Problem 4	58	0.331	1.17×10^{-10}	6	0.043	2.77×10^{-12}
Problem 5	7	0.038	1.55×10^{-15}	7	0.050	2.61×10^{-11}
Problem 6	6	0.033	1.11×10^{-16}	6	0.041	1.55×10^{-11}
Problem 7	6	0.028	6.66×10^{-15}	10	0.046	2.81×10^{-11}
Problem 8	14	0.075	3.47×10^{-10}	10	0.053	4.70×10^{-11}
Problem 9	7	0.033	1.33×10^{-15}	15	0.071	4.11×10^{-11}
Problem 10	9	0.047	2.22×10^{-16}	10	0.054	4.98×10^{-11}
Problem 11	7	0.054	3.13×10^{-13}	8	0.063	3.29×10^{-14}

Table 5.1: Results for the tests run of the LP-Newton and the SMLP-Newton method. The column "Iters" refers to the number of needed iterations and "Time" refers to the elapsed time between start and termination of the algorithm in seconds.

5.1 LP-Newton and SMLP-Newton

In the following, we will analyse the numerical behavior of the LP-Newton method [24] and SMLP-Newton method [53] discussed in Chapter 3. We tested the performance of both algorithms on all the above problems except Problem 2, as the CVXPY solver gets caught in an infinite loop, probably due to numerical problems. The goal of this section is not to determine which of these methods performs "better", as they are suited for different situations. LP-Newton has weaker smoothness assumptions, while SMLP-Newton eliminates the need to calculate derivates, albeit not completely as we will see. We merely want to get an understanding of the convergence characteristics of both algorithms.

In our test runs, we choose $n = 50$ for the variable-length problems. As initial values x_0 for Problems 7–10, we choose the vectors $(1, 0.5)^\top$, $(1, 1, 1.5, 0, 0, 0, 0)^\top$, $(2, 1, 0, 0)^\top$ and $(1, 4, -2, 1, 3, 3, 1, 4, 1, 0, 1, 3, 1, 3)^\top$ respectively, just as in the original papers. For all other problems, we choose the vector $(1, \dots, 1)^\top \in \mathbb{R}^n$ containing only ones. For SMLP-Newton, we initialize $M_0 = F'(x_0)$ and $\eta_0 = \frac{\|F(x_0)\|}{n^2}$, following [53]. Note that we need to calculate a derivative here. This seems necessary for the numerical performance of SMLP-Newton. The runs are terminated as soon as $\|F(x_k)\| < 10^{-10}$.

The results are displayed in Table 5.1. First of all, note that Problems 9 and 10 are nonsmooth at the solution but SMLP-Newton still gives reasonable results. We also need to point out that on Problem 4, LP-Newton performed badly, as the iterations begin to stale. We are unsure why that is, as Problem 4 satisfies the assumptions of LP-Newton. We suspect that this problem arises from numerical inaccuracies. Additionally, SMLP-Newton by chance manages to reaches an exact solution in just one iteration, beating LP-Newton. Other then that, the results look as expected. LP-Newton overall needs fewer iteration to reach the imposed tolerance and also immediately goes well below 10^{-14} with the last iteration for many problems. This is of course due to the quadratic convergence of LP-Newton. SMLP-Newton also manages to solve the problems in a short amount of iterations due to its superlinear convergence.

We will now discuss these convergence rates further. For this, we exemplarily analyse the residuals of Problem 7 via Figure 5.2. From these, we can indeed verify the quadratic

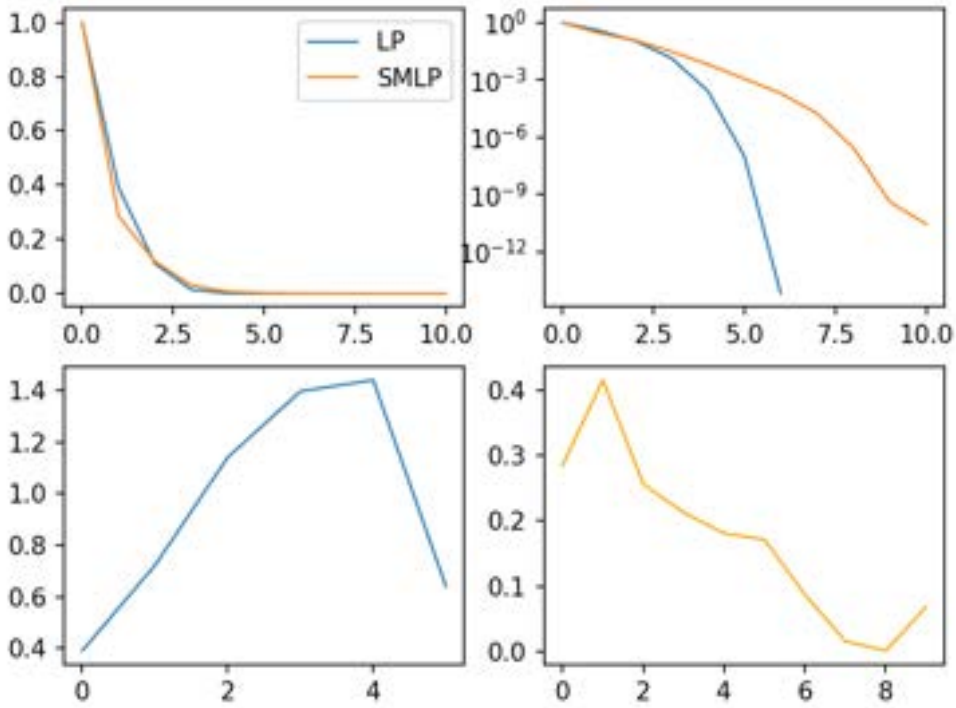

Figure 5.2: Residuals $\|F(x_k)\|$ of LP- and SMLP-Newton on Problem 7 (top), as well as the quotients $\|F(x_{k+1})\|/\|F(x_k)\|^2$ for LP-Newton (bottom left) and the quotients $\|F(x_{k+1})\|/\|F(x_k)\|$ for SMLP-Newton (bottom right).

convergence of LP-Newton and the superlinear convergence of SMLP-Newton. For the former, the quotients $\|F(x_{k+1})\|/\|F(x_k)\|^2$ need to remain bounded, which is indeed the case. For the latter, the quotients $\|F(x_{k+1})\|/\|F(x_k)\|$ need to approach zero but not necessarily monotonically. This is also the case except that quotients go slightly upward towards the end of the iterations for Problem 7, as can be seen in Figure 5.2. This can however be attributed to numerical inaccuracies as a reset of SMLP-Newton (and thus a reset of M_k to the actual Jacobian) at that iterate did not yield better results. Other problems did not show this behavior.

Finally, we want to test Assumption 3 of SMLP-Newton from Section 3.2. To reiterate, the assumption asserts that there are constants $c > 0$ and $0 < \sigma \leq 1$ such that

$$\|N_{k+1} - M_{k+1}\| \leq c\|x_{k+1} - x_k\|^{\sigma}, \tag{5.1}$$

holds, where N_{k+1} is the average Jacobian of F between x_k and x_{k+1}. The authors have acknowledged in their paper, that this assumption does not hold. For this, we analyse the

quotients $\log \|N_{k+1} - M_{k+1}\| / \log \|x_{k+1} - x_k\|$ seen in Figure 5.3. By (5.1), these quotients

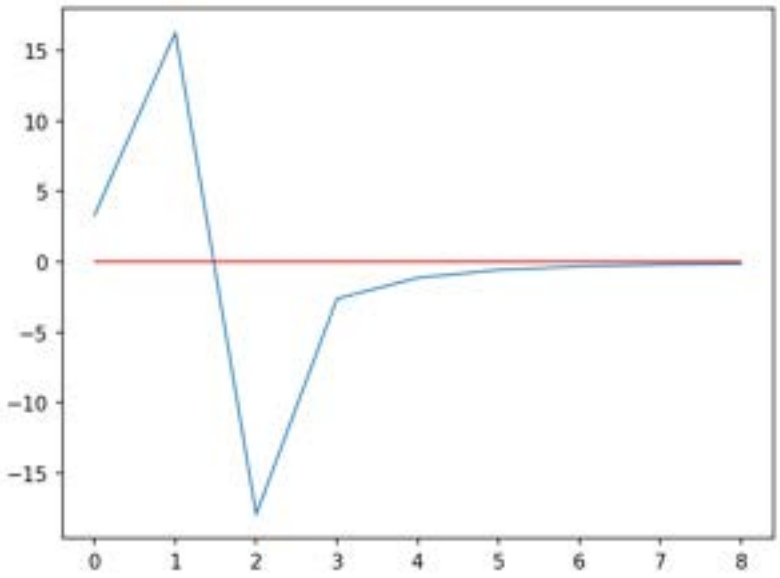

Figure 5.3: Quotients $\log \|N_{k+1} - M_{k+1}\| / \log \|x_{k+1} - x_k\|$ of SMLP-Newton for Problem 11, together with a constant line at zero.

satisfy

$$\frac{\log \|N_{k+1} - M_{k+1}\|}{\log \|x_{k+1} - x_k\|} \geq \sigma + \frac{c}{\log \|x_{k+1} - x_k\|}$$

as soon as $\|x_{k+1} - x_k\| < 1$, as the logarithm then becomes negative. As the fraction on the right approaches zero as $\|x_{k+1} - x_k\|$ approaches zero, the quotients should approach a constant value above σ. However, as the quotients stay below zero, it is to assume that the assumption is violated. This experiment can also be replicated for other problems. Thus, the assumption that the authors needed to prove superlinear convergence does in fact not hold. As we nevertheless see superlinear convergence in the numerics, we suspect that this is due to the initialization of M_0 as the exact Jacobian $F'(x_0)$. The matrices M_k are thus *close enough* to the actual Jacobian $F'(x_k)$ which may result in the superlinear convergence we observe.

We believe that further research is needed to clarify under which conditions we can guarantee the superlinear convergence of SMLP-Newton that applies to the problems tests. This condition may impose stricter conditions on the initial choice of M_k.

5.2 Performance Profiles

In the next section we want to compare the performance of several Conjugate Gradient methods. For this purpose, we use the *performance profiles* introduced by Dolan and Moré [23]. They are a versatile tool in comparing the performance measurements of many solvers on a given set of problems, as they allow visualization in only one plot. Without a good strategy to compare the measurements, evaluating the performance of solvers in a concise way can

be difficult. For example, one could count on how many problems one solver performed the best. This however leaves out a lot of context. How close were the other solver compared to the best? Did one solver beat the other solvers at one problem but then fail miserably at others? Performance profiles alleviate these concerns by evaluating the *ratio* between the measurement for one solver compared to the best measurement on each problem.

Specifically, for a given metric, let $t_{p,s}$ be the measurement from the solver s from the set of solvers S on the problem p from the problem set P. The only requirements on the metric are that its produced values are positive and that lower values are considered better. In the case that solver s did not solve problem p, we set $t_{p,s} = \infty$. Then, the *performance ratios* $r_{p,s}$ for a problem $p \in P$ and a solver $s \in S$ are defined by

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

with the convention that $r_{p,s} = \infty$ whenever $t_{p,s} = \infty$, even if no solver managed to solve problem p resulting in the quotient "∞/∞". Note that this is slightly different to [23], where they instead set $r_{p,s}$ to a high value that is above all the other $r_{p,s}$. This difference is however merely a technicality.

Now, the *performance profile* ρ_s of a solver $s \in S$ is the function $\rho_s \colon [1, \infty) \to [0, 1]$ defined by

$$\rho_s(\tau) = \frac{1}{n_p} \left| \left\{ p \in P \mid r_{p,s} \leq \tau \right\} \right|,$$

where n_p is the number of problems. Thus for given $\tau \geq 1$, the number $\rho_s(\tau)$ denotes the ratio of problems for which the measurement $t_{p,s}$ is within factor τ of the best measurement for this problem p. Clearly, ρ_s is a monotonically increasing and piecewise constant function continuous from the right. We can also see that the number $\rho_s(1)$ denotes the fraction of problems where s performed the best. Also, the number

$$\lim_{\tau \to \infty} \rho_s(\tau)$$

denotes the fraction of problems that s managed to solve.

Thus, performance profiles capture a lot of the performance characteristics of solvers on a given problem set and hence are a major tool in our performance analysis.

5.3 Comparison of Conjugate Gradient Methods

The Conjugate Gradient algorithms tested in this thesis are NDK1 (Ahmed et al. [2]) from Section 4.3.1, MDKM (Waziri et al. [79]), the algorithm from Gao and He [31] from Section 4.3.2, MHZ1 (Sabi'u et al. [67]), GCD (Xiao and Zhu [83]), MFRM (Abubakar et al. [1]) and HSG (Awwal et al. [5]). Algorithms NDK1 and MHZ1 have a "1" in their name, as they are one of two proposed variants of their respective algorithms. We have chosen to only include one of these variants. The variants perform similarly, although the variants tested here performed slightly better in analysis of the original authors. Note that HSG actually is a Spectral Gradient algorithm. We have chosen to include it here, as it also follows the same framework as Algorithm 4.1.

We will run all the monotone test problems for different values of n and different initial values. The initial values are chosen at random from the uniform distribution on $[0, 5]^n$, except for Problem 5 where we sample from $[0, 1]^n$. The algorithms are terminated whenever $\|F_k\| < 10^{-8}$, $\|x_k - x_{k-1}\| < 10^{-14}$ or when the number of iteration exceeds 1000.

In the following tables, "Start" denotes the initial value, "Iters" the number of iterations, "Evals" the number of function evaluation and "Time" the time in seconds elapsed between start and termination of the test run. We will exemplarily display the tables for Problems 1, 3 and 6. We primarily include the tables for Problem 1 as it posed difficulties for most solvers.

		NDK1				MDKM			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	1000	3526	0.063	2.14×10^{-6}	1000	4206	0.070	1.06×10^{-4}
1000	x_1	1000	3588	0.062	1.08×10^{-6}	1000	4214	0.065	2.35×10^{-4}
1000	x_2	12	50	0.001	3.55×10^{-9}	1000	4199	0.065	4.31×10^{-5}
5000	x_0	12	54	0.002	9.17×10^{-10}	1000	4224	0.118	1.85×10^{-5}
5000	x_1	12	54	0.002	9.17×10^{-10}	1000	4243	0.119	8.99×10^{-5}
5000	x_2	12	54	0.002	9.17×10^{-10}	1000	4241	0.119	1.41×10^{-4}
10000	x_0	11	51	0.003	4.02×10^{-9}	1000	4227	0.185	3.08×10^{-5}
10000	x_1	11	51	0.002	4.02×10^{-9}	1000	4231	0.210	1.95×10^{-5}
10000	x_2	11	51	0.002	4.02×10^{-9}	1000	4244	0.207	6.34×10^{-5}
50000	x_0	10	51	0.029	5.15×10^{-10}	1000	4257	2.382	7.65×10^{-6}
50000	x_1	10	51	0.033	5.15×10^{-10}	1000	4279	2.827	1.27×10^{-5}
50000	x_2	10	51	0.033	5.15×10^{-10}	1000	4329	2.778	5.88×10^{-6}
100000	x_0	10	52	0.077	1.01×10^{-9}	1000	4294	6.756	8.04×10^{-6}
100000	x_1	10	52	0.076	1.01×10^{-9}	1000	4299	6.866	7.05×10^{-6}
100000	x_2	10	52	0.076	1.01×10^{-9}	1000	4286	6.801	6.64×10^{-6}
		Gao-He				GCD			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	1000	2544	0.041	1.84×10^{-5}	1000	5127	0.070	1.66×10^{-5}
1000	x_1	1000	2535	0.041	2.90×10^{-5}	1000	5125	0.067	1.62×10^{-5}
1000	x_2	1000	2547	0.041	1.98×10^{-5}	1000	5104	0.068	1.47×10^{-5}
5000	x_0	1000	2542	0.076	1.31×10^{-5}	1000	5160	0.126	7.83×10^{-6}
5000	x_1	1000	2539	0.075	1.43×10^{-5}	1000	5160	0.130	8.45×10^{-6}
5000	x_2	1000	2543	0.075	3.61×10^{-5}	1000	5168	0.125	8.14×10^{-6}
10000	x_0	1000	2527	0.118	1.92×10^{-5}	1000	5199	0.201	4.23×10^{-6}
10000	x_1	1000	2544	0.122	8.02×10^{-6}	1000	5161	0.225	3.98×10^{-6}
10000	x_2	1000	2529	0.132	8.91×10^{-6}	1000	5151	0.219	4.01×10^{-6}
50000	x_0	1000	2547	1.213	2.83×10^{-6}	1000	5292	2.370	3.52×10^{-7}
50000	x_1	1000	2551	1.520	2.04×10^{-6}	1000	5308	3.049	3.55×10^{-7}
50000	x_2	1000	2543	1.455	2.35×10^{-6}	1000	5277	2.951	3.62×10^{-7}
100000	x_0	1000	2570	3.735	1.22×10^{-6}	1000	5305	7.220	6.11×10^{-8}
100000	x_1	1000	2574	4.128	4.59×10^{-7}	1000	5366	7.393	6.11×10^{-8}
100000	x_2	1000	2580	3.834	7.77×10^{-7}	1000	5303	7.225	6.30×10^{-8}

Table 5.4: Results for Problem 1 for solvers NDK1, MDKM, Gao-He and GCD.

		MHZ1				MFRM			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	1000	3156	0.056	4.96×10^{-5}	1000	8125	0.106	2.65×10^{-5}
1000	x_1	1000	3145	0.054	1.95×10^{-5}	1000	8095	0.103	4.21×10^{-4}
1000	x_2	1000	3157	0.055	4.34×10^{-5}	1000	8116	0.105	3.98×10^{-4}
5000	x_0	1000	3148	0.106	7.06×10^{-6}	1000	8135	0.188	1.72×10^{-5}
5000	x_1	1000	3165	0.098	4.08×10^{-5}	1000	8177	0.187	2.02×10^{-4}
5000	x_2	1000	3153	0.101	3.51×10^{-5}	1000	8110	0.186	3.52×10^{-4}
10000	x_0	1000	3156	0.155	1.73×10^{-5}	1000	8136	0.291	1.30×10^{-4}
10000	x_1	1000	3164	0.180	2.01×10^{-5}	1000	8171	0.342	2.37×10^{-4}
10000	x_2	1000	3413	0.183	1.38×10^{-5}	1000	8172	0.334	1.11×10^{-4}
50000	x_0	1000	3144	1.883	1.16×10^{-5}	1000	8160	2.684	1.27×10^{-4}
50000	x_1	1000	3146	2.345	1.18×10^{-5}	1000	8162	3.578	2.31×10^{-4}
50000	x_2	1000	3151	2.430	1.35×10^{-5}	84	791	0.327	2.08×10^{-8}
100000	x_0	1000	3196	6.144	9.05×10^{-6}	1000	8238	7.564	1.81×10^{-4}
100000	x_1	1000	3196	6.525	8.71×10^{-6}	1000	8228	7.702	1.57×10^{-4}
100000	x_2	1000	3196	6.065	8.70×10^{-6}	1000	8233	7.656	1.64×10^{-4}

		HSG			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	1000	44956	0.447	1.60×10^{-5}
1000	x_1	1000	43479	0.425	4.41×10^{-5}
1000	x_2	1000	43432	0.427	4.46×10^{-5}
5000	x_0	1000	42047	0.765	2.10×10^{-5}
5000	x_1	1000	41304	0.749	2.53×10^{-5}
5000	x_2	1000	42369	0.769	2.34×10^{-5}
10000	x_0	1000	40875	1.149	1.40×10^{-5}
10000	x_1	1000	41632	1.194	1.41×10^{-5}
10000	x_2	1000	41293	1.173	1.51×10^{-5}
50000	x_0	1000	39552	11.386	1.77×10^{-6}
50000	x_1	1000	39675	14.226	1.66×10^{-6}
50000	x_2	1000	45500	15.958	3.18×10^{-7}
100000	x_0	1000	38490	28.677	4.15×10^{-7}
100000	x_1	1000	38504	28.958	3.92×10^{-7}
100000	x_2	1000	38178	29.427	3.96×10^{-7}

Table 5.5: Results for Problem 1 for solvers MHZ1, MFRM and HSG.

		NDK1				MDKM			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	9	27	0.001	1.93×10^{-9}	12	25	0.001	2.14×10^{-9}
1000	x_1	9	27	0.001	1.76×10^{-9}	12	25	0.001	1.98×10^{-9}
1000	x_2	9	27	0.001	2.01×10^{-9}	12	25	0.001	2.25×10^{-9}
5000	x_0	9	27	0.001	1.69×10^{-9}	12	25	0.001	1.98×10^{-9}
5000	x_1	9	27	0.001	1.68×10^{-9}	12	25	0.001	1.96×10^{-9}
5000	x_2	9	27	0.001	1.61×10^{-9}	12	25	0.001	1.87×10^{-9}
10000	x_0	9	27	0.002	1.65×10^{-9}	12	25	0.002	1.95×10^{-9}
10000	x_1	9	27	0.002	1.58×10^{-9}	12	25	0.002	1.87×10^{-9}
10000	x_2	9	27	0.002	1.64×10^{-9}	12	25	0.002	1.93×10^{-9}
50000	x_0	9	27	0.036	1.60×10^{-9}	12	25	0.034	1.92×10^{-9}
50000	x_1	9	27	0.036	1.59×10^{-9}	12	25	0.032	1.91×10^{-9}
50000	x_2	9	27	0.036	1.63×10^{-9}	12	25	0.036	1.95×10^{-9}
100000	x_0	9	27	0.076	1.59×10^{-9}	12	25	0.070	1.91×10^{-9}
100000	x_1	9	27	0.079	1.62×10^{-9}	12	25	0.069	1.95×10^{-9}
100000	x_2	9	27	0.077	1.62×10^{-9}	12	25	0.072	1.95×10^{-9}
		Gao-He				GCD			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	10	21	0.000	1.08×10^{-11}	31	122	0.002	5.71×10^{-9}
1000	x_1	9	19	0.000	5.91×10^{-9}	31	122	0.002	5.44×10^{-9}
1000	x_2	10	21	0.000	1.33×10^{-11}	31	122	0.002	5.83×10^{-9}
5000	x_0	10	21	0.001	1.79×10^{-9}	31	122	0.005	5.50×10^{-9}
5000	x_1	10	21	0.001	1.69×10^{-9}	31	122	0.004	5.48×10^{-9}
5000	x_2	10	21	0.001	1.44×10^{-9}	31	122	0.004	5.35×10^{-9}
10000	x_0	11	23	0.002	5.37×10^{-12}	31	122	0.008	5.47×10^{-9}
10000	x_1	11	23	0.002	5.14×10^{-12}	31	122	0.008	5.35×10^{-9}
10000	x_2	11	23	0.002	4.46×10^{-12}	31	122	0.008	5.44×10^{-9}
50000	x_0	11	23	0.031	7.94×10^{-9}	31	122	0.121	5.43×10^{-9}
50000	x_1	11	23	0.032	8.75×10^{-9}	31	122	0.123	5.42×10^{-9}
50000	x_2	11	23	0.030	8.29×10^{-9}	31	122	0.121	5.48×10^{-9}
100000	x_0	12	25	0.059	5.94×10^{-11}	31	122	0.255	5.42×10^{-9}
100000	x_1	12	25	0.054	6.87×10^{-11}	31	122	0.252	5.47×10^{-9}
100000	x_2	12	25	0.050	5.40×10^{-11}	31	122	0.238	5.47×10^{-9}

Table 5.6: Results for Problem 3 for solvers NDK1, MDKM, Gao-He and GCD.

		MHZ1				MFRM			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	23	69	0.001	3.96×10^{-9}	75	199	0.004	3.83×10^{-9}
1000	x_1	23	69	0.001	4.75×10^{-9}	82	222	0.004	9.88×10^{-9}
1000	x_2	23	69	0.001	4.92×10^{-9}	122	333	0.006	3.35×10^{-9}
5000	x_0	23	69	0.003	4.02×10^{-9}	91	249	0.010	6.23×10^{-9}
5000	x_1	22	66	0.003	9.56×10^{-9}	74	216	0.009	8.03×10^{-9}
5000	x_2	22	66	0.003	9.36×10^{-9}	79	224	0.009	6.59×10^{-9}
10000	x_0	23	69	0.005	3.76×10^{-9}	84	238	0.016	8.15×10^{-9}
10000	x_1	22	66	0.005	9.33×10^{-9}	122	341	0.024	9.41×10^{-9}
10000	x_2	23	69	0.005	3.89×10^{-9}	65	181	0.012	9.06×10^{-9}
50000	x_0	22	66	0.069	9.95×10^{-9}	103	278	0.277	6.52×10^{-9}
50000	x_1	22	66	0.071	9.96×10^{-9}	67	185	0.187	7.95×10^{-9}
50000	x_2	22	66	0.068	9.98×10^{-9}	91	250	0.249	6.11×10^{-9}
100000	x_0	22	66	0.157	9.95×10^{-9}	83	233	0.469	1.63×10^{-9}
100000	x_1	22	66	0.166	9.98×10^{-9}	87	245	0.495	7.84×10^{-9}
100000	x_2	23	69	0.150	3.73×10^{-9}	70	199	0.364	9.23×10^{-9}

		HSG			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	14	60	0.001	7.74×10^{-9}
1000	x_1	17	72	0.001	2.38×10^{-9}
1000	x_2	15	64	0.001	1.35×10^{-9}
5000	x_0	14	60	0.003	8.11×10^{-9}
5000	x_1	15	63	0.003	5.25×10^{-9}
5000	x_2	17	72	0.003	2.71×10^{-9}
10000	x_0	14	60	0.005	6.50×10^{-9}
10000	x_1	15	64	0.005	5.97×10^{-9}
10000	x_2	17	69	0.006	7.71×10^{-9}
50000	x_0	15	63	0.069	4.84×10^{-9}
50000	x_1	16	66	0.077	4.09×10^{-9}
50000	x_2	15	63	0.070	4.08×10^{-9}
100000	x_0	15	63	0.136	4.65×10^{-9}
100000	x_1	15	63	0.140	3.93×10^{-9}
100000	x_2	15	63	0.152	4.94×10^{-9}

Table 5.7: Results for Problem 3 for solvers MHZ1, MFRM and HSG.

		NDK1				MDKM			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	11	43	0.001	6.93×10^{-9}	15	64	0.001	8.91×10^{-9}
1000	x_1	11	43	0.001	7.01×10^{-9}	14	62	0.001	8.82×10^{-9}
1000	x_2	11	43	0.001	7.58×10^{-9}	15	64	0.001	8.92×10^{-9}
5000	x_0	11	43	0.002	7.50×10^{-9}	16	69	0.003	1.53×10^{-9}
5000	x_1	11	43	0.002	7.23×10^{-9}	14	62	0.003	8.82×10^{-9}
5000	x_2	11	43	0.002	7.24×10^{-9}	14	62	0.003	8.81×10^{-9}
10000	x_0	11	43	0.003	7.49×10^{-9}	16	69	0.005	1.53×10^{-9}
10000	x_1	11	43	0.004	7.55×10^{-9}	16	69	0.006	1.44×10^{-9}
10000	x_2	11	43	0.003	7.67×10^{-9}	16	69	0.005	1.74×10^{-9}
50000	x_0	11	43	0.053	7.83×10^{-9}	16	69	0.083	1.55×10^{-9}
50000	x_1	11	43	0.056	7.83×10^{-9}	16	69	0.078	1.61×10^{-9}
50000	x_2	11	43	0.056	7.82×10^{-9}	16	69	0.087	1.58×10^{-9}
100000	x_0	11	43	0.122	7.75×10^{-9}	16	69	0.169	1.59×10^{-9}
100000	x_1	11	43	0.124	7.71×10^{-9}	16	69	0.171	1.59×10^{-9}
100000	x_2	11	43	0.118	7.68×10^{-9}	16	69	0.175	1.63×10^{-9}
		Gao-He				GCD			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	21	60	0.001	5.38×10^{-9}	33	163	0.002	6.48×10^{-9}
1000	x_1	20	58	0.001	6.05×10^{-9}	33	163	0.002	6.66×10^{-9}
1000	x_2	21	60	0.001	4.72×10^{-9}	33	163	0.003	6.14×10^{-9}
5000	x_0	20	58	0.002	5.47×10^{-9}	33	163	0.006	6.43×10^{-9}
5000	x_1	20	58	0.003	5.50×10^{-9}	33	163	0.006	6.93×10^{-9}
5000	x_2	20	58	0.003	6.10×10^{-9}	33	163	0.006	7.18×10^{-9}
10000	x_0	20	58	0.004	5.48×10^{-9}	33	163	0.011	6.67×10^{-9}
10000	x_1	20	58	0.005	6.45×10^{-9}	33	163	0.013	7.25×10^{-9}
10000	x_2	20	58	0.005	5.50×10^{-9}	33	163	0.011	6.69×10^{-9}
50000	x_0	19	56	0.073	6.07×10^{-9}	33	163	0.183	6.81×10^{-9}
50000	x_1	19	56	0.065	6.14×10^{-9}	33	163	0.172	6.90×10^{-9}
50000	x_2	19	56	0.078	5.90×10^{-9}	33	163	0.183	6.72×10^{-9}
100000	x_0	19	56	0.154	6.11×10^{-9}	33	163	0.394	6.85×10^{-9}
100000	x_1	19	56	0.146	5.91×10^{-9}	33	163	0.389	6.70×10^{-9}
100000	x_2	19	56	0.152	5.83×10^{-9}	33	163	0.392	6.68×10^{-9}

Table 5.8: Results for Problem 6 for solvers NDK1, MDKM, Gao-He and GCD.

		MHZ1				MFRM			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	20	414	0.005	1.25×10^{-2}	74	617	0.008	9.79×10^{-9}
1000	x_1	34	151	0.002	9.74×10^{-9}	60	496	0.007	9.37×10^{-9}
1000	x_2	27	109	0.002	4.97×10^{-9}	65	545	0.007	7.81×10^{-9}
5000	x_0	29	118	0.005	9.26×10^{-9}	66	549	0.020	9.40×10^{-9}
5000	x_1	37	146	0.006	9.10×10^{-9}	65	539	0.020	8.14×10^{-9}
5000	x_2	29	118	0.005	6.96×10^{-9}	66	549	0.020	9.89×10^{-9}
10000	x_0	39	181	0.014	9.97×10^{-9}	70	585	0.039	9.70×10^{-9}
10000	x_1	36	164	0.015	8.72×10^{-9}	68	567	0.042	5.77×10^{-9}
10000	x_2	30	122	0.010	4.90×10^{-9}	61	511	0.032	9.74×10^{-9}
50000	x_0	35	158	0.187	6.64×10^{-9}	67	565	0.565	4.48×10^{-9}
50000	x_1	38	173	0.201	8.56×10^{-9}	73	613	0.613	9.48×10^{-9}
50000	x_2	41	191	0.232	7.50×10^{-9}	67	565	0.588	8.57×10^{-9}
100000	x_0	37	168	0.426	7.19×10^{-9}	69	578	1.196	7.82×10^{-9}
100000	x_1	29	117	0.299	4.61×10^{-9}	71	595	1.192	7.71×10^{-9}
100000	x_2	32	130	0.333	8.97×10^{-9}	62	520	1.091	9.54×10^{-9}

		HSG			
n	Start	Iters	Evals	Time	$\|F(x^*)\|$
1000	x_0	21	128	0.002	1.17×10^{-9}
1000	x_1	20	140	0.002	4.23×10^{-9}
1000	x_2	24	187	0.003	3.52×10^{-11}
5000	x_0	16	105	0.005	6.22×10^{-11}
5000	x_1	22	155	0.007	2.36×10^{-9}
5000	x_2	23	196	0.009	2.58×10^{-9}
10000	x_0	22	168	0.014	6.47×10^{-12}
10000	x_1	28	222	0.018	2.50×10^{-11}
10000	x_2	21	160	0.014	2.21×10^{-10}
50000	x_0	19	142	0.180	5.56×10^{-10}
50000	x_1	17	106	0.138	1.13×10^{-10}
50000	x_2	22	149	0.177	3.47×10^{-10}
100000	x_0	18	133	0.338	1.79×10^{-10}
100000	x_1	17	109	0.261	9.29×10^{-9}
100000	x_2	21	141	0.351	6.54×10^{-10}

Table 5.9: Results for Problem 6 for solvers MHZ1, MFRM and HSG.

One thing immediately eminent from these profiles is that most solvers perform poorly on Problem 1, while NDK1 performs very well. The only solver that manages to solve Problem 1, except of MFRM in one case, is NDK1 and even NDK1 failed in two cases of the smallest n. This is due to the structure of Problem 1. Let F be the function in Problem 1 and x^* one of its zeros (not necessarily on Ω). With $t^* \coloneqq \sum_{i=1}^{n}(x_i^*)^2$, we get

$$\begin{aligned} 0 = F_i(x^*) &= 2c(x_i^* - 1) + 4(t^* - 0.25)x_i^* \\ &= (4t^* + 2c - 1)x_i^* - 2c, \end{aligned} \tag{5.2}$$

for all $i = 1, \ldots, n$, which implies

$$x_i^* = \frac{2c}{4t^* + 2c - 1}$$

for all $i = 1, \ldots, n$. In particular, all the components of x_i coincide, that is $x_i^* = \bar{x}$ for all $i = 1, \ldots, n$ and an $\bar{x} \in \mathbb{R}$. Hence, $t^* = n\bar{x}^2$ and (5.2) reduces to the polynomial equation

$$0 = 4n\bar{x}^3 + (2c - 1)\bar{x} - 2c.$$

This polynomial, and thus F, has exactly three roots. These roots can be seen in Figure 5.10 for the case $n = 2$. We also see that $\|F(\cdot)\|$ approaches zero around (but not exactly) zero.

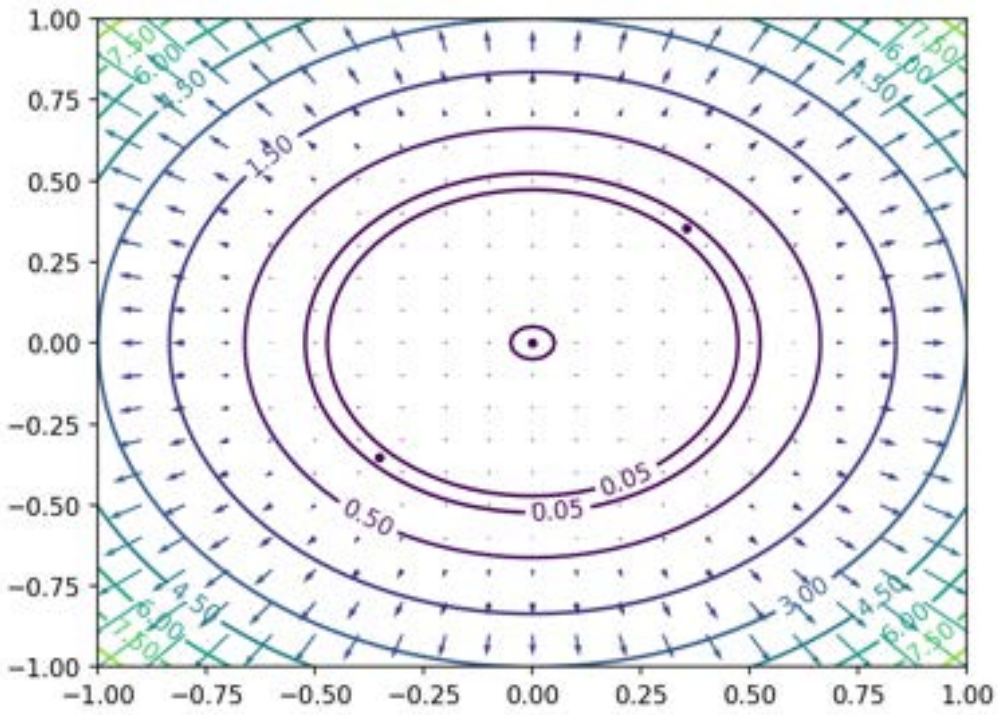

Figure 5.10: Vector field of the function F from Problem 1 for $n = 2$ together with the contours of $\|F(\cdot)\|$. The three points along the diagonal are the zeros of F.

Additionally, there is a ring around zero where $\|F(\cdot)\|$ becomes very small and it also contains the other two zeros. This ring approaches the central root as $n \to \infty$. When an

iterate approaches this ring it can get caught in it and not manage to move across the ring to the other zero or it might not notice that more descent is possible there. Only NDK1 manages to consistently avoid the ring for sufficiently large n.

The numbers from the tables also yield the following performance profiles. Whenever a solver terminated without reaching the desired accuracy of $\|F(x_k)\|$, the run is considered a failure and the respective value in the performance profiles is set to infinity. This results in the inability for all solvers to reach the value 1 in the profiles, as the profiles asymptotically approach the success rate over all tests.

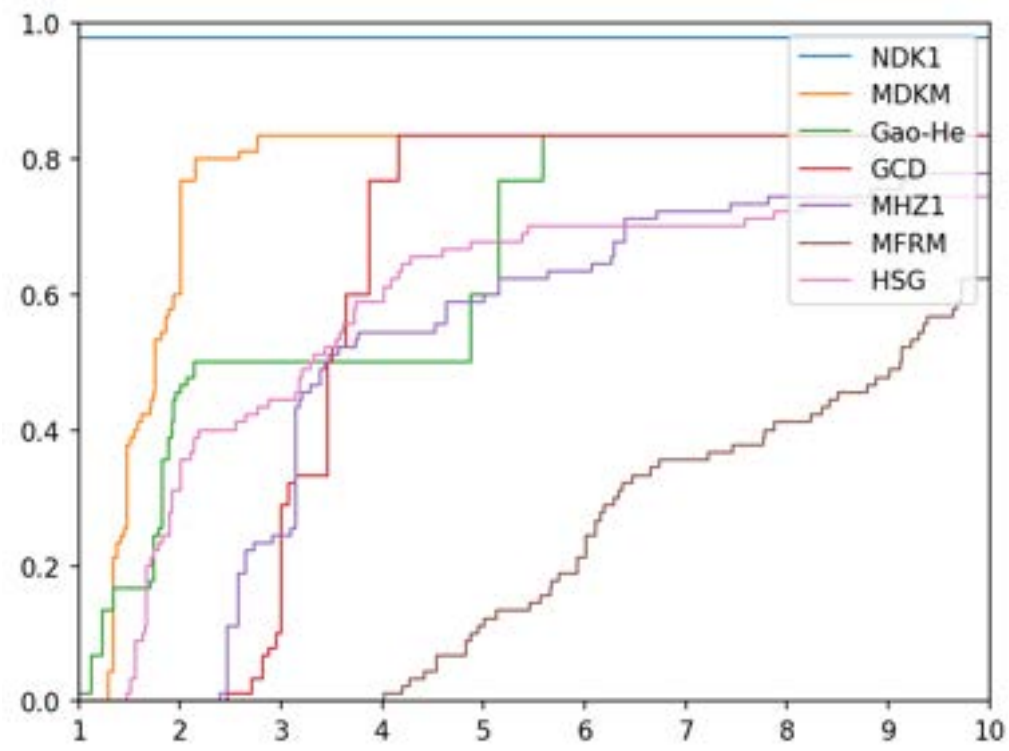

Figure 5.11: Performance profile for the number of iterations.

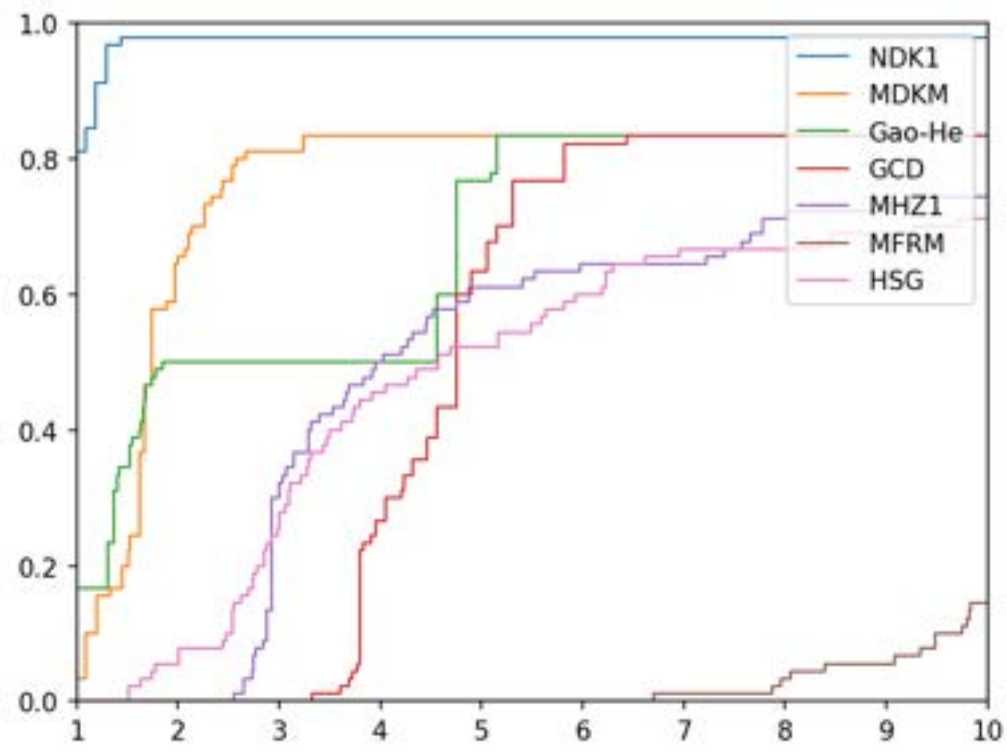

Figure 5.12: Performance profile for the number of function evaluations.

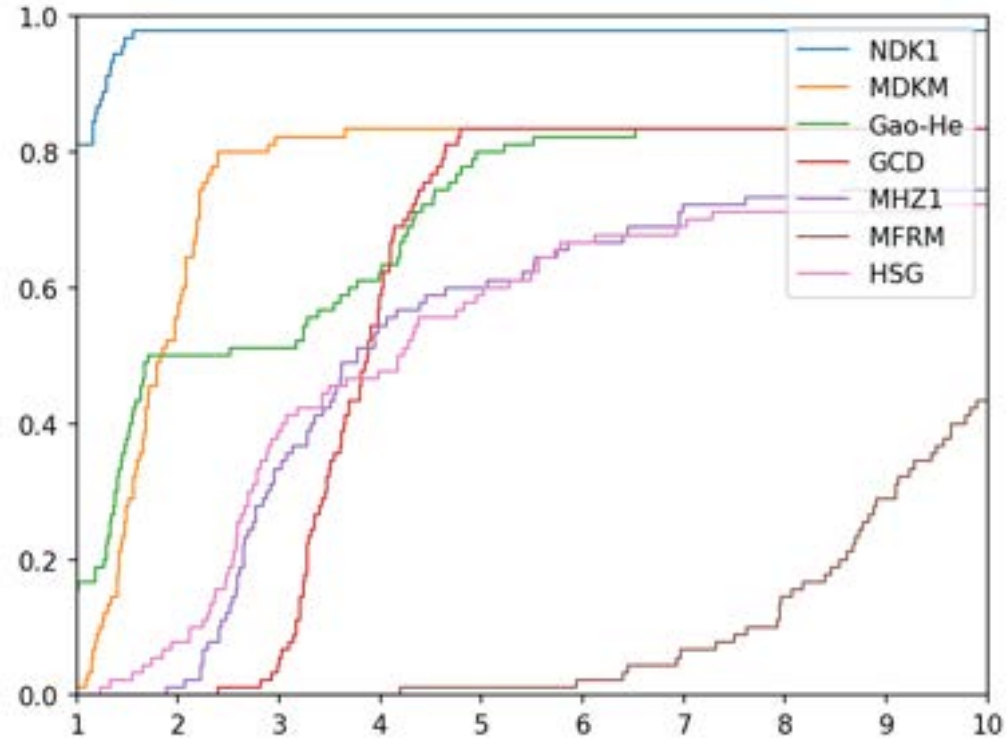

Figure 5.13: Performance profile for the elapsed time.

As we can see, NDK1 significantly outperforms its competitors in all metrics. These results are even more pronounced than in [2]. It always provided the smallest number of iteration and is in other metrics only sometimes beaten by their predecessor MDKM and Gao-He. Also notable is that Gao-He performed very well on some problems but then plateaus, indicating that it may be less consistent compared to the other methods. This plateau however is way shorter in the profile for the elapsed time, which may arguably be the more important metric. Also, unexpectedly, MFRM performed much worse compared to the other methods. We expected MFRM to perform at least competitively, as the benchmarks in [1] suggest. These were however only performed with initial values been various multiples of $(1, \dots, 1)^\top$. Thus, their tests might not have had as broad of a coverage as ours.

We have also analysed the convergence behavior of the residuals in NDK1, MDKM, Gao-He and HSG. For this analysis, we chose $n = 50\,000$ and as initial value the first of the three samples that we used in the prior tests. We choose the lower tolerance $\|F(x_k)\| < 10^{-10}$ to see more of the convergence behavior. The results for Problems 3 and 6 can be found in Figure 5.14. In these plots, we again see that NDK1 is clearly the best performing algorithm among those tested. MDKM and Gao-He perform similarly and HSG takes the longest to reach low residuals. In these plots, all methods seem to converge linearly, as the

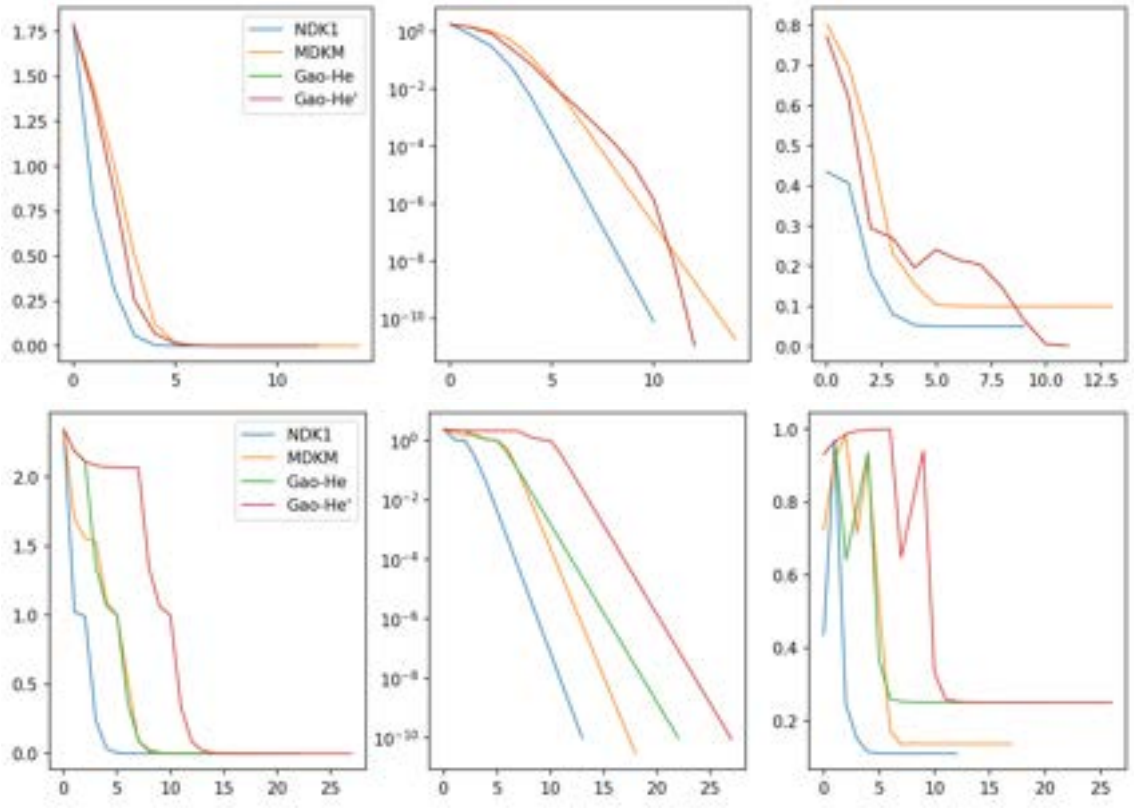

Figure 5.14: Convergence behavior of the NDK1, MDKM, Gao-He and HSG solver for Problem 3 (top) and Problem 6 (bottom). The plots show the residuals $\|F(x_k)\|$ (left and middle), as well as the quotients $\|F(x_{k+1})\|/\|F(x_k)\|$ (right).

quotients all approach values well below one. We also want to highlight how well NDK1 performs generally over all tests. Overall, the residuals $\|F(x_k)\|$ in NDM1 converge with a very low iteration count to tolerances well below 10^{-16} in a linear fashion. Very often, the exponent of the residuals decrease in each step. Thus, it especially is very suited for large scale monotone systems. We are interested in further research whether or not there are schemes other than Dai-Kou schemes that can accomplish these results and what these schemes will have in common.

6

Conclusion

In this thesis, we have numerically and theoretically analysed several methods of the LP-Newton and the Conjugate Gradient type to solve constrained systems of equations. Newton's method, while being very powerful through its fast quadratic convergence rate, requires strong smoothness and regularity assumptions on the function F. In particular, Newton's method does not allow for non-isolated solutions. We have seen that the LP-Newton method successfully addresses these problems and is applicable to constrained systems of equations while still maintaining the convergence rate of Newton's method. The regularity assumptions, in particular, are replaced by the so-called error bound condition that allows for non-isolated solutions. To analyse the impact of the error bound condition on the convergence rate of LP-Newton, we employed a modification that allows for a strengthening and a weakening of this condition. We have shown that LP-Newton still converges under the modified error bound condition and that the modification has a direct impact on the order of convergence.

However, the LP-Newton method is still computationally expensive, as we have to solve a Linear Program at each step of the iteration. Thereby, it is even more expensive than Newton's method in which we need to solve linear systems. Additionally, LP-Newton still requires the calculation of (generalized) Jacobians. To address the last concern, the SMLP-Newton algorithm employs a Quasi-Newton approximation of the Jacobian. For their algorithm, the authors have shown superlinear convergence which we improved to a two-step convergence order. However, to prove superlinear convergence, the authors needed an assumption that is usually violated, as we confirmed numerically. However, we still observed superlinear convergence in the numerics. To find a more realistic assumption that allows to show superlinear convergence of SMLP-Newton, further research is needed.

Conjugate Gradient methods pose another potent class of methods that is also applicable to nonsmooth systems with non-isolated solutions. As they do not use any derivatives and are completely matrix-free, they are suitable to solve large-scale systems. However, they require that the system of equations is monotone. We analysed various Conjugate Gradient algorithms numerically and two of them theoretically. During our analysis, we addressed weaknesses we have discovered in the theory of both methods, including the NDK method, which was clearly the best performing algorithm in our numerical analysis. We observed that the NDK method performed significantly better than other recent algorithms; it out-

performed the other methods on almost all problems. Additionally, the authors have shown linear convergence under the error bound condition, which is clearly visible in the numerics. Over all of our numerical analysis, the two algorithms that performed best are both of the Dai-Kou class. Thus, the question arises whether there are algorithms for other classes of CG methods that provide similar strong results. This allows for further research.

Bibliography

[1] Abubakar, A. B., Kumam, P., Mohammad, H., Awwal, A. M., and Sitthithakerngkiet, K. A Modified Fletcher–Reeves Conjugate Gradient Method for Monotone Nonlinear Equations with Some Applications. In: *Mathematics* 7(8):745, 2019. DOI: 10.3390/math7080745.

[2] Ahmed, K., Waziri, M. Y., Halilu, A. S., and Murtala, S. On two symmetric Dai-Kou type schemes for constrained monotone equations with image recovery application. In: *EURO Journal on Computational Optimization* 11:100057, 2023. DOI: 10.1016/j.ejco.2023.100057.

[3] Andrei, N. Open Problems in Nonlinear Conjugate Gradient Algorithms for Unconstrained Optimization. In: *Bulletin of the Malaysian Mathematical Sciences Society. Second Series* 34(2):319–330, 2011.

[4] Andrei, N. et al. *Nonlinear conjugate gradient methods for unconstrained optimization*. Springer, 2020. DOI: 10.1007/978-3-030-42950-8.

[5] Awwal, A. M., Kumam, P., Abubakar, A. B., Wakili, A., and Pakkaranang, N. A New Hybrid Spectral Gradient Projection Method for Monotone System Equations with Convex Constraints. In: *Thai Journal of Mathematics* 125:147, 2018. ISSN: 1686-0209.

[6] Barzilai, J. and Borwein, J. M. Two-Point Step Size Gradient Methods. In: *IMA journal of numerical analysis* 8(1):141–148, 1988. DOI: 10.1093/imanum/8.1.141.

[7] Becher, L., Fernández, D., and Ramos, A. A trust-region LP-Newton method for constrained nonsmooth equations under Hölder metric subregularity. In: *Computational Optimization and Applications*:1–33, 2023. DOI: 10.1007/s10589-023-00498-9.

[8] Bogle, I. D. L. and Perkins, J. D. A New Sparsity Preserving Quasi-Newton Update for Solving Nonlinear Equations. In: *SIAM journal on scientific and statistical computing* 11(4):621–630, 1990. DOI: 10.1137/0911036.

[9] Broyden, C. G. A class of methods for solving nonlinear simultaneous equations. In: *Mathematics of computation* 19(92):577–593, 1965. DOI: 10.1090/s0025-5718-1965-0198670-6.

[10] Broyden, C. G. The convergence of an algorithm for solving sparse nonlinear systems. In: *Mathematics of Computation* 25(114):285–294, 1971. DOI: 10.1090/s0025-5718-1971-0297122-5.

[11] Broyden, C. G., Dennis Jr, J. E., and Moré, J. J. On the Local and Superlinear Convergence of Quasi-Newton Methods. In: *IMA Journal of Applied Mathematics* 12(3):223–245, 1973. DOI: 10.1093/imamat/12.3.223.

[12] Bubeck, S. Theory of Convex Optimization for Machine Learning. In: *ArXiv* abs/1405.4980, 2014.

[13] Cauchy, A. Méthode générale pour la résolution des systemes d'équations simultanées. In: *Comp. Rend. Sci. Paris* 25(1847):536–538, 1847.

[14] Cheng, W. A PRP type method for systems of monotone equations. In: *Mathematical and Computer Modelling* 50(1-2):15–20, 2009. DOI: 10.1016/j.mcm.2009.04.007.

[15] Clarke, F. H. *Optimization and Nonsmooth Analysis*. SIAM, 1990. ISBN: 9781611971309. DOI: 10.1137/1.9781611971309.

[16] Dai, Y.-H. and Kou, C.-X. A Nonlinear Conjugate Gradient Algorithm with an Optimal Property and an Improved Wolfe Line Search. In: *SIAM Journal on Optimization* 23(1):296–320, 2013. DOI: 10.1137/100813026.

[17] Dai, Y.-H. and Liao, L.-Z. New conjugacy conditions and related nonlinear conjugate gradient methods. In: *Applied Mathematics and optimization* 43:87–101, 2001. DOI: 10.1007/s002450010019.

[18] Dai, Y.-H. and Yuan, Y. A Nonlinear Conjugate Gradient Method with a Strong Global Convergence Property. In: *SIAM Journal on Optimization* 10(1):177–182, 1999. DOI: 10.1137/s1052623497318992.

[19] Dembo, R. S., Eisenstat, S. C., and Steihaug, T. Inexact Newton Methods. In: *SIAM Journal on Numerical analysis* 19(2):400–408, 1982. DOI: 10.1137/0719025.

[20] Dennis Jr, J. E. and Schnabel, R. B. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. SIAM, 1996. DOI: 10.1137/1.9781611971200.

[21] Ding, Y., Xiao, Y., and Li, J. A class of conjugate gradient methods for convex constrained monotone equations. In: *Optimization* 66(12):2309–2328, 2017. DOI: 10.1080/02331934.2017.1372438.

[22] Dirkse, S. P. and Ferris, M. C. MCPLIB: A collection of nonlinear mixed complementarity problems. In: *Optimization methods and software* 5(4):319–345, 1995. DOI: 10.1080/10556789508805619.

[23] Dolan, E. D. and Moré, J. J. Benchmarking Optimization Software with Performance Profiles. In: *Mathematical Programming* 91:201–213, 2002. DOI: 10.1007/s101070100263.

[24] Facchinei, F., Fischer, A., and Herrich, M. An LP-Newton method: nonsmooth equations, KKT systems, and nonisolated solutions. In: *Mathematical Programming* 146:1–36, 2014. DOI: 10.1007/s10107-013-0676-6.

[25] Facchinei, F. and Kanzow, C. A nonsmooth inexact Newton method for the solution of large-scale nonlinear complementarity problems. In: *Mathematical Programming* 76(3):493–512, 1997. DOI: 10.1007/bf02614395.

[26] Figueiredo, M. A., Nowak, R. D., and Wright, S. J. Gradient Projection for Sparse Reconstruction: Application to Compressed Sensing and Other Inverse Problems. In: *IEEE Journal of selected topics in signal processing* 1(4):586–597, 2007. DOI: 10.1109/JSTSP.2007.910281.

[27] Fischer, A. Solution of monotone complementarity problems with locally Lipschitzian functions. In: *Mathematical Programming* 76:513–532, 1997. DOI: 10.1007/BF02614396.

[28] Fischer, A., Herrich, M., Izmailov, A. F., and Solodov, M. V. A Globally Convergent LP-Newton Method. In: *SIAM Journal on Optimization* 26(4):2012–2033, 2016. DOI: 10.1137/15M105241X.

[29] Fletcher, R. *Practical methods of optimization*. John Wiley & Sons, 2000. DOI: 10.1002/9781118723203.

[30] Fletcher, R. and Reeves, C. M. Function minimization by conjugate gradients. In: *The computer journal* 7(2):149–154, 1964. DOI: 10.1093/comjnl/7.2.149.

[31] Gao, P. and He, C. An efficient three-term conjugate gradient method for nonlinear monotone equations with convex constraints. In: *Calcolo* 55:1–17, 2018. DOI: 10.1007/s10092-018-0291-2.

[32] Geiger, C. and Kanzow, C. *Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben*. Springer, 1999. ISBN: 9783642585821. DOI: 10.1007/978-3-642-58582-1.

[33] Geiger, C. and Kanzow, C. *Theorie und Numerik restringierter Optimierungsaufgaben*. Springer, 2002. ISBN: 9783642560040. DOI: 10.1007/978-3-642-56004-0.

[34] Hager, W. W. and Zhang, H. A New Conjugate Gradient Method with Guaranteed Descent and an Efficient Line Search. In: *SIAM Journal on optimization* 16(1):170–192, 2005. DOI: 10.1137/030601880.

[35] Hager, W. W. and Zhang, H. A survey of nonlinear conjugate gradient methods. In: *Pacific journal of Optimization* 2(1):35–58, 2006.

[36] Hestenes, M. R. and Stiefel, E. Methods of conjugate gradients for solving linear systems. In: *Journal of Research of the National Bureau of Standards* 49(6):409, 1952. DOI: 10.6028/jres.049.044.

[37] Kelley, C. T. *Iterative Methods for Linear and Nonlinear Equations*. SIAM, 1995. DOI: 10.1137/1.9781611970944.

[38] Klatte, D. and Kummer, B. *Nonsmooth Equations in Optimization: Regularity, Calculus, Methods and Applications*. Vol. 60. Nonconvex Optimization and Its Applications. Springer, 2005. DOI: 10.1007/b130810.

[39] Königsberger, K. *Analysis 2*. Springer, 2000. DOI: 10.1007/978-3-662-05702-5.

[40] La Cruz, W. A projected derivative-free algorithm for nonlinear equations with convex constraints. In: *Optimization Methods and Software* 29(1):24–41, 2014. DOI: 10.1080/10556788.2012.721129.

[41] La Cruz, W., Martínez, J., and Raydan, M. Spectral residual method without gradient information for solving large-scale nonlinear systems of equations. In: *Mathematics of computation* 75(255):1429–1448, 2006. DOI: 10.1090/S0025-5718-06-01840-0.

[42] La Cruz, W., Martínez, J. M., and Raydan, M. *Spectral residual method without gradient information for solving large-scale nonlinear systems: Theory and experiments*. Technical Report RT-04-08. Dpto. de Computacion, UCV, 2004.

[43] La Cruz, W. and Raydan, M. Nonmonotone Spectral Methods for Large-Scale Nonlinear Systems. In: *Optimization Methods and software* 18(5):583–599, 2003. DOI: 10.1080/10556780310001610493.

[44] Li, D.-H. and Fukushima, M. A derivative-free line search and global convergence of Broyden-like method for nonlinear equations. In: *Optimization methods and software* 13(3):181–201, 2000. DOI: 10.1080/10556780008805782.

[45] Li, D.-H. and Fukushima, M. A modified BFGS method and its global convergence in nonconvex minimization. In: *Journal of Computational and Applied Mathematics* 129(1-2):15–35, 2001. DOI: 10.1016/s0377-0427(00)00540-9.

[46] Li, D.-H. and Wang, X.-L. A modified Fletcher-Reeves-type derivative-free method for symmetric nonlinear equations. In: *Numer. Algebra Control Optim* 1(1):71–82, 2011. DOI: 10.3934/NACO.2011.1.71.

[47] Liu, J.-K. and Li, S.-J. A projection method for convex constrained monotone nonlinear equations with applications. In: *Computers & Mathematics with Applications* 70(10):2442–2453, 2015. DOI: 10.1016/j.camwa.2015.09.014.

[48] Liu, Y and Storey, C Efficient generalized conjugate gradient algorithms, part 1: theory. In: *Journal of optimization theory and applications* 69:129–137, 1991. DOI: 10.1007/bf00940464.

[49] Marini, L., Morini, B., and Porcelli, M. Quasi-Newton methods for constrained nonlinear systems: complexity analysis and applications. In: *Computational Optimization and Applications* 71:147–170, 2018.

[50] Martınez, J. M. Practical quasi-Newton methods for solving nonlinear systems. In: *Journal of computational and Applied Mathematics* 124(1-2):97–121, 2000. DOI: 10.1016/s0377-0427(00)00434-9.

[51] Martinez, J. M. and Zambaldi, M. C. An inverse column-updating method for solving large–scale nonlinear systems of equations. In: *Dynamical Systems* 1(2):129–140, 1992. DOI: 10.1080/10556789208805512.

[52] Martínez, M. d. l. Á. and Fernández, D. A quasi-Newton modified LP-Newton method. In: *Optimization Methods and Software* 34(3):634–649, 2019. DOI: 10.1080/10556788.2017.1384955.

[53] Martínez, M. d. l. Á. and Fernández, D. On the Local and Superlinear Convergence of a Secant Modified Linear-Programming-Newton Method. In: *Journal of Optimization Theory and Applications* 180:993–1010, 2019. DOI: 10.1007/s10957-018-1407-1.

[54] Meintjes, K. and Morgan, A. P. A methodology for solving chemical equilibrium systems. In: *Applied Mathematics and Computation* 22(4):333–361, 1987. DOI: 10.1016/0096-3003(87)90076-2.

[55] Morini, B., Porcelli, M., and Toint, P. L. Approximate norm descent methods for constrained nonlinear systems. In: *Mathematics of Computation* 87(311):1327–1351, 2018. DOI: 10.1090/mcom/3251.

[56] Nocedal, J. and Wright, S. J. *Numerical Optimization*. Springer, 2006.

[57] Oren, S. S. and Spedicato, E. Optimal conditioning of self-scaling variable metric algorithms. In: *Mathematical programming* 10(1):70–90, 1976. DOI: 10.1007/bf01580654.

[58] Ortega, J. M. and Rheinboldt, W. C. *Iterative solution of nonlinear equations in several variables*. SIAM, 2000. DOI: 10.1137/1.9780898719468.

[59] Perry, A. *A Class of Conjugate Gradient Algorithms with a Two-Step Variable Metric Memory*. Discussion Paper 269. Evanston, IL, 1977. URL: http://hdl.handle.net/10419/220629.

[60] Perry, A. A Modified Conjugate Gradient Algorithm. In: *Operations Research* 26(6):1073–1078, 1978. DOI: 10.1287/opre.26.6.1073.

[61] Polak, E. and Ribière, G. Note sur la convergence de méthodes de directions conjuguées. In: *Revue française d'informatique et de recherche opérationnelle. Série rouge* 3(16):35–43, 1969. DOI: 10.1051/m2an/196903r100351.

[62] Polyak, B. T. The conjugate gradient method in extremal problems. In: *USSR Computational Mathematics and Mathematical Physics* 9(4):94–112, 1969. DOI: 10.1016/0041-5553(69)90035-4.

[63] Pozrikidis, C. *An introduction to Grids, Graphs, and Networks*. Oxford University Press, USA, 2014. ISBN: 9780199996728.

[64] Qi, L. and Jiang, H. Semismooth Karush-Kuhn-Tucker Equations and Convergence Analysis of Newton and Quasi-Newton Methods for Solving these Equations. In: *Mathematics of Operations Research* 22(2):301–325, 1997. DOI: 10.1287/moor.22.2.301.

[65] Qi, L. and Sun, J. A nonsmooth version of Newton's method. In: *Mathematical Programming* 58(1–3):353–367, 1993. DOI: 10.1007/bf01581275.

[66] Rockafellar, R. T. and Wets, R. J.-B. *Variational Analysis*. Vol. 317. Springer, 1998. DOI: 10.1007/978-3-642-02431-3.

[67] Sabi'u, J., Shah, A., and Waziri, M. Y. A modified Hager-Zhang conjugate gradient method with optimal choices for solving monotone nonlinear equations. In: *International Journal of Computer Mathematics* 99(2):332–354, 2022. DOI: 10.1080/00207160.2021.1910814.

[68] Schubert, L. K. Modification of a quasi-Newton method for nonlinear equations with a sparse Jacobian. In: *Mathematics of Computation* 24(109):27–30, 1970. DOI: 10.1090/s0025-5718-1970-0258276-9.

[69] Shanno, D. F. Conjugate Gradient Methods with Inexact Searches. In: *Mathematics of Operations Research* 3(3):244–256, 1978. DOI: 10.1287/moor.3.3.244.

[70] Shanno, D. F. On the Convergence of a New Conjugate Gradient Algorithm. In: *SIAM Journal on Numerical Analysis* 15(6):1247–1257, 1978. DOI: 10.1137/0715085.

[71] Solodov, M. V. and Svaiter, B. F. A globally convergent inexact Newton method for systems of monotone equations. In: *Reformulation: Nonsmooth, piecewise smooth, semismooth and smoothing methods*:355–369, 1999. DOI: 10.1007/978-1-4757-6388-1_18.

[72] Sun, W. and Yuan, Y.-X. *Optimization Theory and Methods: Nonlinear Programming*. Vol. 1. Springer Science & Business Media, 2006. DOI: 10.1007/b106451.

[73] Ulbrich, M. *Semismooth Newton methods for variational inequalities and constrained optimization problems in function spaces*. SIAM, 2011. DOI: 10.1137/1.9781611970692.

[74] Ulbrich, M. and Ulbrich, S. *Nichtlineare Optimierung*. Birkhäuser Basel, 2012. DOI: 10.1007/978-3-0346-0654-7.

[75] Voigts, J. URL: https://github.com/johannesvoigtsuzl/constrained-systems.

[76] Wang, C. and Wang, Y. A superlinearly convergent projection method for constrained systems of nonlinear equations. In: *Journal of Global Optimization* 44:283–296, 2009. DOI: 10.1007/s10898-008-9324-8.

[77] Wang, C., Wang, Y., and Xu, C. A projection method for a system of nonlinear monotone equations with convex constraints. In: *Mathematical Methods of Operations Research* 66:33–46, 2007. DOI: 10.1007/s00186-006-0140-y.

[78] Waziri, M. Y. and Ahmed, K. Two descent Dai-Yuan conjugate gradient methods for systems of monotone nonlinear equations. In: *Journal of Scientific Computing* 90:1–53, 2022. DOI: 10.1007/s10915-021-01713-7.

[79] Waziri, M. Y., Ahmed, K., and Halilu, A. S. A modified Dai–Kou-type method with applications to signal reconstruction and blurred image restoration. In: *Computational and Applied Mathematics* 41(6):232, 2022. DOI: 10.1007/s40314-022-01917-z.

[80] Wolfe, P. Convergence conditions for ascent methods. In: *SIAM review* 11(2):226–235, 1969. DOI: 10.1137/1011036.

[81] Wolfe, P. Convergence conditions for ascent methods. II: Some corrections. In: *SIAM review* 13(2):185–188, 1971. DOI: 10.1137/1013035.

[82] Wood, A. J., Wollenberg, B. F., and Sheblé, G. B. *Power Generation, Operation, and Control*. John Wiley & Sons, 1996. DOI: 10.1016/0140-6701(96)88715-7.

[83] Xiao, Y. and Zhu, H. A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing. In: *Journal of Mathematical Analysis and Applications* 405(1):310–319, 2013. DOI: 10.1016/j.jmaa.2013.04.017.

[84] Zhang, L., Zhou, W., and Li, D. Some descent three-term conjugate gradient methods and their global convergence. In: *Optimisation Methods and Software* 22(4):697–711, 2007. DOI: 10.1080/10556780701223293.