# An Infeasible-Path-Following Algorithm For Nonlinear Multiobjective Optimisation Problems

by

Philipp Alexander Naegele

A thesis submitted to
The University of Birmingham
for the degree of
Doctor of Philosophy

School of Mathematics
The University of Birmingham
October 2009

# Abstract

The subject area of multiobjective optimisation deals with the investigation of optimisation problems that possess more than one objective function. Usually, there does not exist a single solution that optimises all functions simultaneously, quite the contrary, in general the set of so-called efficient points, these are solutions to multiobjective optimisation problems, is large. Since it is important for the decision maker to obtain as much information as possible about this set, our research objective is to determine a well-defined and meaningful approximation of the solution set for nonlinear multiobjective optimisation problems.

In order to achieve this target we develop an algorithm that employs the optimality conditions introduced by Karush, Kuhn and Tucker for a scalarised objective function and computes solutions to the corresponding system of equations via a modified Newton method. In particular, we utilise an infeasible interior-point technique which determines solutions in the neighbourhood of a central path and therefore, constitutes a path-following approach. We proof the convergence of our algorithm under certain assumptions and develop a warm-start strategy to compute different solutions for varying weighting parameters.

Furthermore we examine our numerical implementation in MATLAB and present the results we obtained for several suites of test problems from the literature.

# Acknowledgements

I would like to thank my supervisors Dr Sándor Zoltán Németh and Prof Dr Jörg Fliege for their invaluable advice and support during the investigation of the ideas presented in this thesis.

Moreover, I want to thank my parents and my sisters Annette and Susanne for their incredible support and encouragement. Not only during my time in Birmingham, but in all those years!

”Ich sage nur ein Wort: ”Vielen Dank!”” (Horst Hrubesch)

In addition, I am very grateful to all the friends who turned my time in Birmingham into such a great experience. In particular, special thanks goes to Lucy J. Adamson, the boys of the ’Birmingham Blue Jays’ baseball team, the members of the mighty Postgrad Mathletic football team and last but definitely not least the ’motley crew’ of office 315.

Finally, I am grateful for the financial assistance provided by the ESPRC and the School of Mathematics.

# Contents

# Chapter 1

# Introduction

Multiobjective optimisation examines problems featuring several different objective functions which have to be considered simultaneously. Usually these objectives are also competing with each other which aggravates the situation. For instance, a fundamental challenge in portfolio management is to maximise the rate of return while reducing the risk involved in the investment(s) at the same time. In general there does not exist a single solution that optimises all given functions of the problem at once. Thus, a selection of alternatives, so-called efficient solutions, has to be determined: a set of feasible solutions for which there do not exist different feasible solutions with the same or better objective function values that possess a strictly better function value in at least one criterion. Typically, the set of efficient solutions is very large and the different solutions are incomparable with each other with respect to the employed ordering concept. The reason for the incomparability is that we utilise partial orders in vector spaces rather than total orders. A more detailed examination of this subject area is presented in Section 2.1.2 in the next chapter. Hence, it is vital to provide the decision maker with as much information about the solution set as possible for him to gain crucial insight into the problem and its structure. The computation of well-defined approximations of the solution sets for nonlinear multiobjective optimisation problems is the subject of this thesis.

In practice, numerous applications from a wide range of subject areas benefit from multiobjective optimisation. Prominent examples emerge from management science [10], [2], [17], [37], [47], engineering design [24], [15], [40], environmental analysis [12] or cancer treatment planning [22], [9].

The theoretical foundations for multiobjective optimisation problems were first established by Vilfredo Pareto (1848-1923) [36] and Francis Y. Edgeworth (1845-1926) [7] at the end of the nineteenth century. Their motivation to consider several different objectives originated from economics, most notably from the context of welfare theory and utility theory. They introduced a concept of optimality for a multi-agent utility problem and constituted a general notion of an economic equilibrium. In addition, they investigated and specified conditions which, if satisfied, assure the existence of such equilibria. However, their findings were based strongly on assumptions of the existence of utility functions. In the first half of the past century Zorn's Lemma [51] then paved the way to identify conditions for the existence of minimal elements in preordered sets without relying on utility functions. Finally, the seminal results of Harold W. Kuhn and Albert W. Tucker [28] regarding necessary and sufficient conditions for solutions to nonlinear optimisation problems in the 1950s represented another landmark in this subject area and initiated further research and development of this field. For further information on the historical background we refer the reader to the sound survey paper by Stadler [41].

The determination of efficient solutions and efficient sets characterises an essential part of multiobjective optimisation. In conjunction with this task, a crucial aspect that needs thorough analysis is the comparability of elements under consideration. For instance, a total order entails serious drawbacks in the context of solving multiobjective optimisation problems which is discussed more thoroughly in Fishburn [11, Chapters 7 and 8]. Thus, typically a partial order is employed which consequently serves as basis for the introduction of an efficiency notion. Another important point that requires careful attention is the existence of efficient sets and their inherent properties. Vital in this analysis is the concept of convex sets and functions.

In this thesis we compute efficient points by employing a scalarisation approach. That is, we solve ersatz problems in scalarised form which are obtained by weighting and summating the single objective functions. Each substitute problem is characterised by an unique weighting parameter vector, the entries of which are the specific weights for the individual objective functions. Solutions to these problems constitute so-called properly efficient points with respect to our initial multiobjective optimisation problem. Under certain prerequisites these properly efficient points are dense within the efficient set and consequently, these sets are identical from a numerical point of view. The advantage of this approach is that these scalarised problems can be solved efficiently by well-known algorithms, for example by interior-point methods. In this thesis we utilise a special variant of these techniques, namely an infeasible interior-point algorithm. That is, starting from an initial point we compute a series of points that converges to an efficient solution. The crucial advantage of the infeasible approach is that the iterates do not have to satisfy the side constraints of the optimisation problem, which drastically simplifies the computation of admissible starting points and iterates. Furthermore, we pursue a path-following strategy which restricts these iterates to a special neighbourhood of the so-called central path. That way the iterates are guided in a sophistical manner to an optimal solution and moreover, their infeasibility is gradually reduced. As a result, a sufficiently large number of solved scalarised problems should provide the decision maker with a useful approximation of the efficient set.

A fundamental challenge lies in the selection of the 'right' weighting parameters that lead to different optimal solutions that subsequently constitute meaningful approximations of

the efficient sets. In order to attack this issue we utilise a warm-start strategy. The quintessence of this technique is that current iterates of the interior-point method for one particular scalarised ersatz problem instance are perturbed to create different iterates which correspond to different ersatz problem with different weighting parameters. Since we commence with this progress immediately in our algorithm we obtain broad approximations of the efficient set at early stages of the computation. Consequently, we are in a position to identify regions of the weighting parameter space that promise to produce different efficient solutions and can shift the emphasis of our efforts to address these areas during the generation of the approximation. Besides interactively detecting suitable weighting parameters, this strategy also reduces the computational effort since we exploit information from advanced iterates rather than repeatedly starting the solution process from the same initial point.
Altogether, combining all theoretical concepts outlined above into one algorithm, we are generating sequences of iterates that converge to optimal solutions of the original multiobjective optimisation problem and form an approximation of the efficient set.

Scalarisation methods represent the most common route to solve multiobjective optimisation problems. A sound survey of the most important variations of these techniques can be found in Hillermeier [19, Chapter 3.2]. Besides, stochastic methods and evolutionary approaches as well as different deterministic optimisation techniques such as branch and bound algorithms for example are other established strategies for solving multiobjective optimisation problems.
In the 1980s interior-point methods were developed to design algorithms with more satisfying theoretical properties than the well-known 'Simplex Method' devised by George B. Dantzig which was previously unrivalled in the framework of linear programming. In 1979 the 'Ellipsoid Method' proposed by Leonid G. Khachiyan [26] was the first algorithm

to solve linear programming problems featuring polynomial time complexity. However, since this technique usually approaches its worst-case bound on complexity, it did not prove to be competitive in practical applications. Narendra K. Karmarkar [25] presented his 'Projective Algorithm' in 1984, which also possesses polynomial time complexity, but additionally showed good performance in practice. His revolutionary method ignited a vast amount of research in this newly created subject area and represented the starting point for the design of plenty other interior-point algorithms. One key factor for the efficient operation of Karmarkar's algorithm is that it avoids the boundary of the feasible set by employing a barrier function approach. This lead to the concept of the central path for optimisation problems, which is the set of minima of these auxiliary barrier functions. In 1986 James Renegar derived the first path-following algorithm for linear programming problems, which computes iterates in the vicinity of the central path that continue along it to an optimal solution. This strategy is also utilised in this thesis. In our case, we employ a less stringent neighbourhood which essentially requires the points to solely satisfy a non-negativity condition.
Our warm-start strategy to determine new iterates for different instances of the scalarised ersatz problems is inspired by the the paper by E. Alper Yildirim and Stephen J. Wright [49]. Their work dealt with linear programming, while Heermann [18] extended their ideas for convex quadratic multiobjective optimisation problems.

To complement the theoretical results and research findings we develop a numerical implementation of our solution method in a MATLAB environment. In addition to a description of the computer code we provide a thorough examination of the interactions between the several subroutines and specify the inherent parameters of our algorithm. Furthermore, we formulate a test suite of multiobjective optimisation problems to analyse and evaluate the performance of our computer programme.

We proceed in this thesis as follows. In Chapter 2 we discuss the fundamentals of multiobjective optimisation. The connection between ordering relations and ordering cones is established and based on these concepts we constitute an efficiency notion. In addition, we examine important properties of efficient sets and by introducing a more restrictive efficiency definition we derive conditions under which the set of efficient solutions can be computed by solving scalarised ersatz problems.

Chapter 3 sets the stage for the formulation of an infeasible interior-point path-following algorithm to solve nonlinear multiobjective optimisation problems. Initially, we deduce the corresponding system of Karush-Kuhn-Tucker conditions and present a modified Newton method to solve it. Furthermore, we investigate the central path in details and prove its existence and uniqueness.

In Chapter 4 we present expedient adjustments to the Newton system and establish our path-following algorithm. Moreover, we constitute assumptions that are required to proof that our solution method converges to an optimal solution. These are followed by the actual proof of convergence.

Chapter 5 examines the warm-start strategy to determine new iterates for different weighting parameters and concludes the investigation of the theoretical concepts underlying our solution method.

In Chapter 6 we detail the aspects of the numerical implementation of our algorithm in MATLAB and specify the numerous parameters utilised by the several subroutines of the computer code.

Chapter 7 investigates our collection of test problems and their original sources. Furthermore, we categorise the optimisation problems into different classes and provide graphical illustrations of our results as well as performance measures regarding the computational complexity.

Lastly, in Chapter 8 we conclude our entire research findings and indicate potential areas of interest for future research.

# Chapter 2

# Multiobjective Optimisation

In this chapter we investigate optimisation problems which feature more than one objective function. In the beginning fundamental concepts that are essential for multiobjective optimisation are established. That is, ordering cones are introduced and key properties of efficient sets are examined in detail. Subsequently, the existence of efficient points and techniques to compute these are analysed. One method to determine solutions to multiobjective problems is to consider related scalarised problems, which leads to the definition of properly efficient points. Finally the major result, that the properly efficient points are dense within the set of efficient solutions, is presented at the end of this chapter.

The following discussion of the crucial aspects of multiobjective optimisation is based on the monographs by Ehrgott [8, Chapters 1–3] and Göpfert & Nehse [16, Chapter 2] as well as the Master thesis by Heermann [18, Chapter 2].

## 2.1 General formulation of a multiobjective optimisation problem

As a starting point, we establish a general formulation of the problem. Let $\Omega$ be a nonempty subset of a vector space, $k > 1$ and $f_1, \ldots, f_k$ be functions $f_i : \Omega \to \mathbb{R}$ for

$i = 1, \ldots, k$. Then the general multiobjective optimisation problem is defined as

$$\begin{aligned} \text{"min"} \quad & f(x) = (f_1(x), \ldots, f_k(x)) \\ \text{s.t.} \quad & x \in \Omega. \end{aligned} \tag{2.1}$$

We call the set of alternatives $\Omega$ the **feasible set** of problem (2.1). The space containing the feasible set $\Omega$ is denoted as **decision space**, whereas the space that contains the image of the feasible set $f(\Omega)$ is referred to as **criterion space**.

The minimising-operator was deliberately written in quotation marks since it proves to be essential that we specify in details what we mean by this particular minimisation. For the moment the general idea of multiobjective minimsation is that several objective functions have to be minimised simultaneously.

We proceed with a brief review of the definitions of convex sets and functions followed by an examination of orders and cones.

### 2.1.1 Convex definitions

**Definition 2.1 (Convex set)** *A set $S \subset \mathbb{R}^n$ is denoted as **convex** if*

$$\alpha z^1 + (1-\alpha) z^2 \ \in S$$

*for all $z^1, z^2 \in S$ and for all $\alpha \in [0,1]$.*

**Definition 2.2 (Convex function)** *Let $S \subset \mathbb{R}^n$ be a nonempty convex set. A function $f : S \to \mathbb{R}$ is denoted as **convex** if*

$$f(\alpha z^1 + (1-\alpha) z^2) \leqslant \alpha f(z^1) + (1-\alpha) f(z^2) \tag{2.2}$$

*is satisfied for all $z^1, z^2 \in S$ and for all $\alpha \in [0,1]$.*
*Moreover, a function $f$ is said to be* ***strictly convex*** *if (2.2) holds as a strict inequality for $z^1 \neq z^2$ and for all $\alpha \in (0,1)$.*

*Furthermore, assume $f_1, \dots, f_k$ to be functions $f_i : S \to \mathbb{R}$ for all $i = 1, \dots, k$. A function $f(x) := (f_1(x), \dots, f_k(x))$ is labelled* ***convex*** *if the functions $f_i$ are convex for all $i = 1, \dots, k$.*

In addition, we outline useful properties of convex functions in the next theorem.

**Theorem 2.1** *Let $f_1, \dots, f_k$ be convex functions $f_i : \mathbb{R}^n \to \mathbb{R}$ for $i = 1, \dots, k$ and let $\omega_1, \dots, \omega_k > 0$. Then the function*

$$f(x) := \sum_{i=1}^{k} \omega_i f_i(x)$$

*is convex.*

*Also convex functions of the type $f : \mathbb{R}^n \to \mathbb{R}$ are continuous.*

*Proof.* For the first statement see Hiriart-Urruty & Lemaréchal [20, page 158]. For the second one consult Rockafellar [38, page 83]. $\square$

### 2.1.2 Orders and cones

In order to increase the legibility of this text we employ the subsequent notation. Let $S, S_1, S_2 \subset \mathbb{R}^k$ and $\beta \in \mathbb{R}$. The multiplication of a scalar with a set is indicated by

$$\beta S := \{\beta z \mid z \in S\},$$

in particular $-S = \{-z \mid z \in S\}$. Moreover, the algebraic sum of two sets is defined as

$$S_1 + S_2 := \{z^1 + z^2 \mid z^1 \in S_1, z^2 \in S_2\}.$$

In case $S_1 = \{z^1\}$ is a singleton we use the form $z^1 + S_2$ instead of $\{z^1\} + S_2$.

The next concept that we introduce is a binary relation on a given set.

**Definition 2.3 (Binary relation)** *Let $S$ be any set. A **binary relation** $\mathcal{R}$ on $S$ is a subset of $S \times S$. It is called*

- ***reflexive**, if $(z, z) \in \mathcal{R}$ for all $z \in S$,*
- ***transitive**, if $(z^1, z^2) \in \mathcal{R}$ and $(z^2, z^3) \in \mathcal{R} \Longrightarrow (z^1, z^3) \in \mathcal{R}$ for all $z^1, z^2, z^3 \in S$,*
- ***antisymmetric**, if $(z^1, z^2) \in \mathcal{R}$ and $(z^2, z^1) \in \mathcal{R} \Longrightarrow z^1 = z^2$ for all $z^1, z^2 \in S$,*
- ***total** if $(z^1, z^2) \in \mathcal{R}$ or $(z^2, z^1) \in \mathcal{R}$ for all $z^1, z^2 \in S$.*

*A relation is named a **partial order** if it is reflexive, transitive and antisymmetric. If $\mathcal{R}$ solely satisfies the first two properties it is denoted as a **preorder** (or quasi-order).*

*Let us now consider relations on $\mathbb{R}^k$. A binary relation $\mathcal{R}$ is said to be **compatible with scalar multiplication** if $(\beta z^1, \beta z^2) \in \mathcal{R}$ holds for all $(z^1, z^2) \in \mathcal{R}$ and for all $\beta \in \mathbb{R}, \beta > 0$. Furthermore, the relation $\mathcal{R}$ is labelled as **compatible with addition** if $(z^1 + x, z^2 + x) \in \mathcal{R}$ is true for all $(z^1, z^2) \in \mathcal{R}$ and for all $x \in \mathbb{R}^k$.*

In the following we make use of a more convenient notation for binary relations, namely the symbol $\preceq$. For instance, the expression $(x, y) \in \mathcal{R}$ is replaced by $x \preceq y$ and $(x, y) \notin \mathcal{R}$ is substituted by $x \npreceq y$, respectively.
To exemplify the concept of a binary relation we present two prominent orders on the $\mathbb{R}^n$.

**Example 2.1** *Let $x, y \in \mathbb{R}^k$ and define $l^*(x,y) := \min\{i \mid x_i \neq y_i\}$ for $x \neq y$. A **componentwise order** and a **lexicographic order**, both on the $\mathbb{R}^k$, can be introduced by the following binary relations*

$$x \leqslant y \;:\Longleftrightarrow\; x_i \leqslant y_i \quad \forall i = 1, \ldots, k,$$

$$x \leqslant_{lex} y \;:\Longleftrightarrow\; x_{l^*(x,y)} < y_{l^*(x,y)} \text{ or } x = y.$$

*We observe that $\leqslant$ represents a partial order, while $\leqslant_{lex}$ satisfies the conditions of a total order.*

We can now progress to illustrate how a particular subset of $\mathbb{R}^k$ can be exploited to interpret properties of orders geometrically. It proves to be extremely beneficial that we can take advantage of these two equivalent perspectives on orders. This concept relies on the definition of a cone which we establish next.

**Definition 2.4 (Cone)** *A subset $C \subseteq \mathbb{R}^k$ is named a **cone** if $\beta c \in C$ for all $c \in C$ and for all $\beta \in \mathbb{R}, \beta > 0$. A cone is referred to as*

- ***nontrivial** (or proper), if $C \neq \emptyset$ and $C \neq \mathbb{R}^k$,*
- ***convex**, if $\alpha c^1 + (1-\alpha)c^2 \in C$ for all $c^1, c^2 \in C$ and for all $\alpha \in [0,1]$,*
- ***pointed** (or said to contain no lines), if $C \cap (-C) \subseteq \{0\}$.*

**Theorem 2.2** *A cone $C$ is convex if and only if it is closed under addition. In other words,*

$$\alpha c^1 + (1-\alpha)c^2 \in C \quad \begin{array}{l} for\ all\ c^1, c^2 \in C \\ and\ for\ all\ \alpha \in [0,1] \end{array} \quad \Longleftrightarrow \quad c^1 + c^2 \in C\ for\ all\ c^1, c^2 \in C.$$

*Proof.* Firstly, assume that the cone C is convex. Then we can conclude for $c^1, c^2 \in C$ and $\alpha = 0.5$ in combination with the cone property of $C$ that

$$\begin{aligned} 0.5c^1 + (1-0.5)c^2 &\in C \\ 2(0.5c^1 + (1-0.5)c^2) &\in C \\ c^1 + c^2 &\in C. \end{aligned}$$

Secondly, suppose that the cone $C$ is closed under addition. Exploiting the fact that $C$ is a cone we deduce for $c^1, c^2 \in C$ and $\alpha \in [0,1]$ that $\alpha c^1 \in C$ and $(1-\alpha)c^2 \in C$. Furthermore, since the cone is closed under addition we derive for these elements that

$$\alpha c^1 + (1-\alpha)c^2 \in C.$$ □

**Remark 2.1** *Combining the definition of a cone and the above theorem we can state that $C \subseteq \mathbb{R}^k$ is a convex cone if $\beta C \subseteq C$ for all $\beta > 0$ and $C + C \subseteq C$.*

With respect to a given order relation $\preceq$ on $\mathbb{R}^k$ let us now define the set

$$C_{\preceq} := \{y - x \mid x \preceq y\}. \tag{2.3}$$

**Theorem 2.3** *Let $\preceq$ be a relation which is compatible with scalar multiplication. Then $C_{\preceq}$ defined in (2.3) is a cone.*

*Proof.* Compare Ehrgott [8, page 13–14]. □

The following result constitutes how certain properties of a binary relation $\preceq$ affect the characteristics of the cone $C_{\preceq}$.

**Theorem 2.4** *Let $\preceq$ be a binary relation on $\mathbb{R}^k$ which is compatible with scalar multiplication and addition and let $C_\preceq$ be as defined in (2.3). Then the following statements are true.*

- $0 \in C_\preceq$ *if and only if $\preceq$ is reflexive.*
- $C_\preceq$ *is convex if and only if $\preceq$ is transitive.*
- $C_\preceq$ *is pointed if and only if $\preceq$ is antisymmetric.*

*Proof.* See Ehrgott [8, pages 14–15]. □

Conversely, we can use a cone to formulate an order relation which is compatible with scalar multiplication and addition in $\mathbb{R}^k$. Let $C$ be a cone. We define a binary relation $\preceq_C$ by

$$x \preceq_C y \Longleftrightarrow y - x \in C. \tag{2.4}$$

**Theorem 2.5** *Let $C \subset \mathbb{R}^k$ be a cone. Then $\preceq_C$ as defined in (2.4) is compatible with scalar multiplication and addition in $\mathbb{R}^k$ and the following statements are satisfied.*

- $\preceq_C$ *is reflexive if and only if $0 \in C$.*
- $\preceq_C$ *is transitive if and only if $C$ is convex.*
- $\preceq_C$ *is antisymmetric if and only if $C$ is pointed.*

*Proof.* Consult Ehrgott [8, pages 15–16]. □

Theorems 2.4 and 2.5 establish an equivalence of certain ordering relations and cones. These results enable us to examine multiobjective optimisation problems geometrically. To illustrate this connection let us formulate the corresponding ordering cones to the orders introduced in Example 2.1.

**Example 2.2** *With regards to the componentwise order $\leqslant$ on the $\mathbb{R}^k$ the related ordering cone is the non-negative orthant*

$$C_{\leqslant} := \mathbb{R}^k_+ = \{x \in \mathbb{R}^k \mid x_i \geqslant 0, i = 1, \ldots, k\},$$

*which is called* ***standard ordering cone****.*
*Concerning the lexicographic order $\leqslant_{lex}$ on the $\mathbb{R}^k$ the associated ordering cone can be defined as*

$$\begin{aligned} C_{\leqslant_{lex}} := \{x \in \mathbb{R}^k \mid \; & x_i = 0 \textit{ for all } i = 1, \ldots, k \textit{ or} \\ & x_1 > 0 \textit{ or} \\ & x_1 = \ldots = x_{i-1} = 0 \textit{ and } x_i > 0 \textit{ for } i = 1, \ldots, k\}. \end{aligned}$$

The standard ordering cone represents an especially important ordering cone on the $\mathbb{R}^k$ and we employ this particular cone in the following discussion, unless otherwise stated.

**Remark 2.2** *We highlight that total orders which also feature the antisymmetric property are induced by cones that are not closed [14, page 212]. Consequently, this correlation causes difficulties in the context of numerical methods and hence, our focus lies on partial orders in the subsequent analysis.*

## 2.2 Properties and existence of the efficient set

In this section we introduce an efficiency concept that was established by Edgeworth and Pareto and subsequently present an existence result for the set of efficient solutions. For simplicity and legibility we use the notation $S = f(\Omega)$ for the image of the feasible set $\Omega$ and $z = f(x)$ for $x \in \Omega$ in the following examination. Hence, the optimisation problem

(2.1) can be restated as

$$\begin{aligned} \text{"min"} \quad & z \\ \text{s.t.} \quad & z \in S. \end{aligned} \tag{2.5}$$

**Definition 2.5 (Efficiency via order relations)** *An element $z^* \in S$ is denoted as* ***efficient****, if there exists no other element $z \in S$, that differs from $z^*$ and is less than it with respect to the employed ordering relation $\preceq_C$. That is,*

$$\nexists z \in S : z \neq z^*, z \preceq_C z^*.$$

In reference to the analogous geometrical approach to orders we give an equivalent definition of efficient points using the concept of ordering cones.

**Definition 2.6 (Efficiency via ordering cones)** *Let $S$ be a set and $C$ a related ordering cone. Then $z^* \in S$ is called* ***efficient****, if there exists no $z \in S$ such that*

$$z^* - z \in C \backslash \{0\}.$$

*Moreover, all efficient elements for problem (2.5) are combined into the* ***efficient set***

$$E(S, C) := \{z \in S \mid S \cap (z - C) = \{z\}\}.$$

*In addition, if $S$ is the image of the feasible set $\Omega$, $x^* \in \Omega$ is labelled as* ***Pareto optimal*** *if $z^* \in S$ satisfying $z^* = f(x^*)$ is efficient.*

The following theorems outline distinct properties of efficient sets. We start with the observation that the feasible region can be enhanced by the ordering cone without affecting the efficient set, which is illustrated in Figure 2.1.

**Theorem 2.6** *$E(S,C) = E(S + C,C)$.*

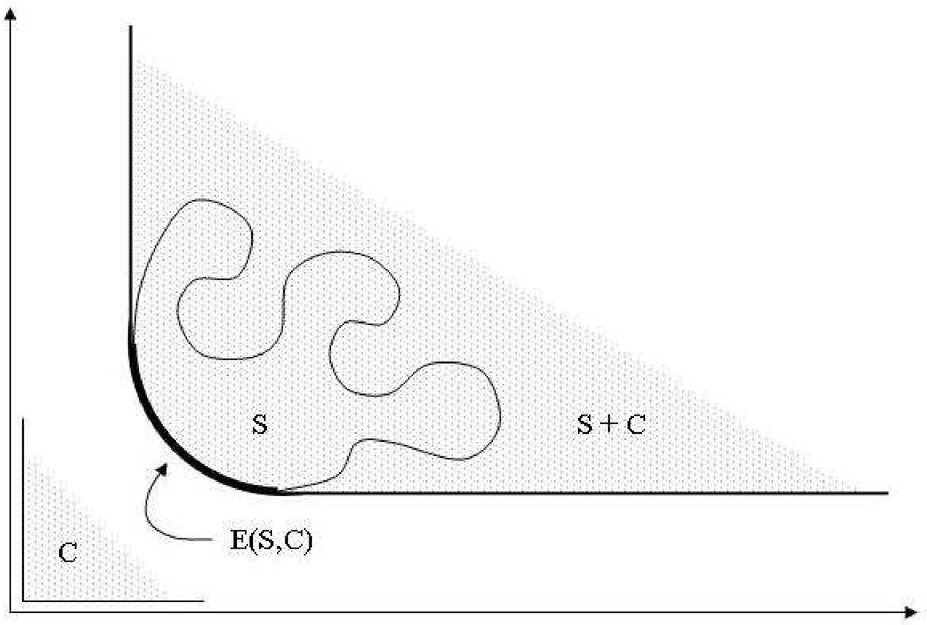


Figure 2.1: Enhancing the feasible set $S$ by the ordering cone $C$ does not change the efficient set.

*Proof.* Compare Ehrgott [8, page 27]. □

Another characteristic of efficient points is that they must belong to the boundary of the feasible set.

**Theorem 2.7** *$E(S, C) \subseteq \partial(S)$.*

*Proof.* See Ehrgott [8, page 28] or Göpfert & Nehse [16, page 31]. □

Before a result concerning the connectedness of the efficient set can be stated, we recall the definition of a connected set.

**Definition 2.7 (Connectedness)** *A set $S \subset \mathbb{R}^k$ is said to be* ***connected*** *if there do not exist two open sets $O_1, O_2 \subset \mathbb{R}^k$ such that $S \subseteq O_1 \cup O_2$ and*

$$S \cap O_1 \neq \emptyset, \; S \cap O_2 \neq \emptyset, \; S \cap O_1 \cap O_2 = \emptyset.$$

**Theorem 2.8** *If $S$ is a closed and convex set and the section $(z - C) \cap S$ is compact for all $z \in S$, then $E(S, C)$ is connected.*

*Proof.* Consult Ehrgott [8, pages 88–89]. $\square$

To establish an existence result for the efficient set we need to introduce an essential proposition, namely Zorn's lemma. It utilises the idea of an inductive order which itself employs the concept of a lower bound. These two important notions are presented in the subsequent definitions.

**Definition 2.8 (Lower bound)** *Let $(S, \preceq)$ be a preordered set, i.e. $\preceq$ is reflexive and transitive, and $A \subset S$ be a subset of $S$. Then an element $z \in S$ is a **lower bound** for $A$ if $z \preceq a$ for all $a \in A$.*

**Definition 2.9 (Inductive order)** *Let $(S, \preceq)$ be a preordered set. Then $(S, \preceq)$ is denoted as **inductively ordered** if every totally ordered subset of $(S, \preceq)$, also referred to as a **chain**, has a lower bound.*

**Theorem 2.9 (Zorn's lemma)** *Let the preordered set $(S, \preceq)$ be inductively ordered. Then $S$ contains a minimal element $z^* \in S$ such that*

$$z \preceq z^* \Longrightarrow z^* \preceq z.$$

*Proof.* We refer the reader to [27] for a derivation from the axiom of choice. $\square$

With this fundamental result at our disposal we are able to prove an existence theorem for the efficient set $E(S, C)$.

**Theorem 2.10** *Let $S$ be a nonempty set and let $C$ be a closed cone. Suppose there exists a $z^0 \in S$ such that the set $S^0 := \{z \in S \mid z \leqslant z^0\} = (z^0 - C) \cap S$ is compact. Then $E(S, C)$ is nonempty. A graphical illustration is given in Figure 2.2.*

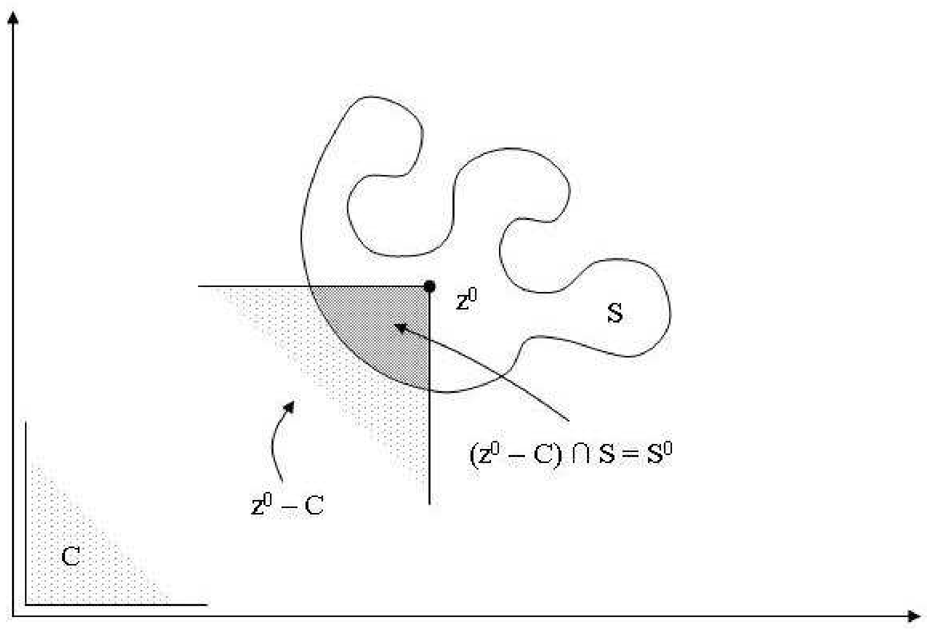


Figure 2.2: The efficient set is nonempty if the set $S^0$ is compact.

*Proof.* Initially, we provide a brief outline of the proof. In the beginning the compactness of $S^0$ is exploited to derive that there exists a lower bound for every chain in $S^0$. Hence, $S^0$ is inductively ordered and we can employ Zorn's lemma to deduce that $S^0$ contains a minimal element $z^*$. Finally, we prove that this element $z^*$ is efficient with regard to $S$ by contradiction.

Let $S^0$ be the assumed compact set and let $S^{\mathcal{J}} := \{z^j \mid j \in \mathcal{J}\}$ be a chain in $S^0$, where $\mathcal{J}$ is an arbitrary index set. In addition, let $\mathcal{I} := \{I \subset \mathcal{J} \mid card(I) < \infty\}$ be the set of all finite subsets of $\mathcal{J}$. Since $S^{\mathcal{J}}$ is a chain in $S^0$ and all $I \in \mathcal{I}$ are finite there exist $z^I := \inf\{z^i \mid i \in I\}$ for all $I \in \mathcal{I}$. Notice that $z^I \in S^0$ is satisfied for all $I \in \mathcal{I}$.

Consider the sets $S^j := (z^j - C) \cap S^0$ for all $j \in \mathcal{J}$. These sets are compact since they are closed subsets of the compact set $S^0$. Moreover, the intersections of all $S^j$ for $j \in I \in \mathcal{I}$ are nonempty since they contain at least $z^I$. Subsequently, due to the compactness of $S^0$

we obtain that the intersection of all $S^j$ for $j \in \mathcal{J}$ is nonempty, i.e. there exists a

$$\widehat{z} \in \bigcap_{j \in \mathcal{J}} (z^j - C) \cap S^0.$$

In other words, $\widehat{z} \leqslant_C z^j$ for all $j \in \mathcal{J}$ and consequently $\widehat{z} \in S^0$ establishes a lower bound for the chain $S^{\mathcal{J}}$. Thus, the chain is inductively ordered and we can utilise Zorn's lemma to guarantee the existence of a minimal element $z^* \in S^0$.
Eventually, suppose $z^* \notin E(S, C)$. Then there exists a $\tilde{z} \in S$, $\tilde{z} \neq z^*$ such that

$$\begin{aligned} \tilde{z} \ \in \ (z^* - C) \cap S \ &\subset \ [(z^0 - C) \cap S - C] \cap S \\ &\subset \ (z^0 - C) \cap S - C \\ &= \ S^0 - C. \end{aligned}$$

Clearly, this contradicts the minimality of $z^*$, which completes the proof. □

We recapitulate that we have proven the existence of efficient elements and the connectedness of the efficient set under certain prerequisites. However, in practical applications it is usually too expensive to compute the efficient set entirely. Hence, we need to employ appropriate approximation techniques. In this thesis we concentrate on the concept of scalarisation to solve multiobjective optimisation problems. But before we can investigate this method in further details we need to review some underlying ideas and introduce a modified definition of efficiency in the next section.

## 2.3 Scalarisation and proper efficiency

In the given context scalarisation is considered to be a technique that transforms a multiobjective optimisation problem into a family of single-criterion ones. This method employs scalarising functionals and we show in the following discussion that certain elements induce such functionals if they belong to a distinct set, namely the dual cone. In addition,

we derive that the solutions of the single-criterion problems are dense in the efficient set under certain convexity assumptions.
We proceed by establishing the definition of a dual cone.

**Definition 2.10 (Dual cone)** *Let $C \subset \mathbb{R}^k$ be a cone. The* ***dual cone*** *$C^*$ is defined by*

$$C^* := \{c^* \in \mathbb{R}^k \mid (c^*)^T c \geqslant 0 \quad \forall c \in C\}.$$

*The set*

$$C^+ := \{c_+ \in \mathbb{R}^k \mid (c_+)^T c > 0 \quad \forall c \in C \backslash \{0\}\}$$

*is named the* ***quasi-interior*** *of the dual cone.*

**Remark 2.3** *Regarding the properties of these sets we note that the case $C^+ = \emptyset$ can occur. For example, consider $C := \{(x, y) \in \mathbb{R}^2 \mid x \geqslant 0\}$. We deduce that $C^* = \{(x, 0) \in \mathbb{R}^2 \mid x \geqslant 0\}$ and moreover, $C^+ = \emptyset$.*
*Furthermore, we can easily derive that $C^+ = int(C^*)$ if $int(C^*) \neq \emptyset$.*

**Remark 2.4** *The dual cone $C^*$ is a closed and convex cone since the scalar product is linear in $c^*$ and the limit of every convergent sequence in $C^*$ is also an element of $C^*$. Göpfert & Nehse [16, pages 26–27] state criteria for which the cones $C$ and $C^*$ are pointed. If the non-negative orthant $C := \mathbb{R}^k_+$ is used as the ordering cone then the dual cone is $C^* := \mathbb{R}^k_+$. Hence, we notice that both cones are pointed.*

Before we can apply the scalarisation concept mentioned above we need to secure that the employed functionals comply with the ordering relation that evaluates feasible solutions. Therefore, we formulate a monotonicity criteria which these functionals must satisfy.

**Definition 2.11 (C-monotonicity)** *Let $\preceq_C$ and $\leqslant$ be relations on $\mathbb{R}^k$ and $\mathbb{R}$, respectively. Furthermore, let $\varphi$ be a function $\varphi : \mathbb{R}^k \to \mathbb{R}$. Then $\varphi$ is named* ***C-monotone***

*with respect to $\preceq_C$ if*

$$x \preceq_C y \Longrightarrow \varphi(x) \leqslant \varphi(y)$$

*for all $x, y \in \mathbb{R}^k$. Moreover, the function $\varphi$ is called **strictly C-monotone** with respect to $\preceq_C$ if*

$$x \preceq_C y, x \neq y \Longrightarrow \varphi(x) < \varphi(y)$$

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

**Remark 2.5** *The elements $c_+ \in C^+$ generate strictly C-monotone functionals $\varphi(z) := c_+^T z$. That is, the functionals created by these elements preserve the order established by the ordering cone.*

We are now prepared to introduce the idea of proper efficiency which selects special elements of the efficient set and thus, constitutes a more restrictive efficiency definition.

**Definition 2.12 (Proper efficiency)** *Let $C \subseteq \mathbb{R}^k$ be a cone and $S \subseteq \mathbb{R}^k, S \neq \emptyset$ a set. Then $z^* \in S$ is denoted as **properly efficient** with respect to problem (2.5) if there exists a $c_+ \in C^+$ such that*

$$c_+^T z^* \leqslant c_+^T z \quad \text{for all } z \in S.$$

*The set of all properly efficient elements with respect to (2.5) is denoted by $P(S, C)$.*

By definition all elements in the efficient set $E(S, C)$ maintain a special interrelation. Efficient solutions necessarily have to deteriorate in at least one objective function in order to remain in the efficient set if they are improved in another criterion.
In certain instances these trade-offs between particular objectives can be unbounded. Properly efficient points are precisely these efficient points which feature solely bounded trade-offs. This concept is further demonstrated in the subsequent example.

**Example 2.3 (Ehrgott [8])** *Let the non-negative orthant be the employed ordering cone* $C = \mathbb{R}^2_+$ *and define the image of the feasible set* $\Omega$ *for an optimisation problem of the form (2.5) as*

$$S = \{z = (z_1, z_2) \in \mathbb{R}^2 \mid (z_1 - 1)^2 + (z_2 - 1)^2 \leqslant 1 \ , \ 0 \leqslant z_1, z_2 \leqslant 1\}.$$

*This example is depicted in Figure 2.3.*

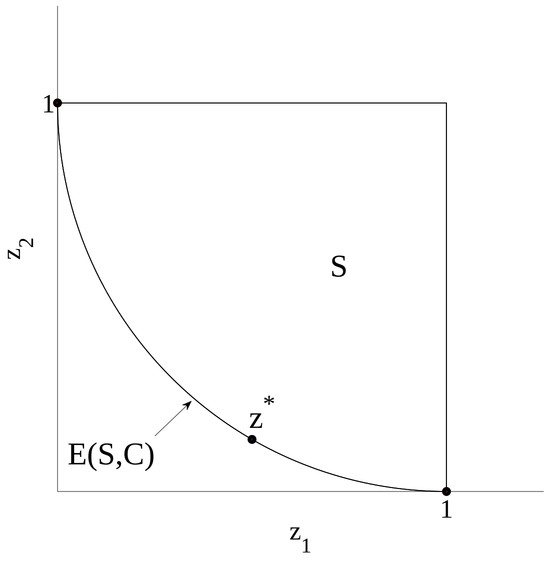


Figure 2.3: Example analysing properly efficient points.

*Obviously the efficient set is constituted by* $E(S, C) = \{(z_1, z_2) \in S \mid (z_1 - 1)^2 + (z_2 - 1)^2 = 1\}$ *and* $z^*$ *as well as* $(1, 0)$ *represent efficient points. Furthermore, we notice that if we shift* $z^*$ *along the efficient line towards* $(1, 0)$ *the increase of* $z_1^*$ *necessarily has to become more substantial the more we decrease the* $z_2^*$ *component. Hence, in the limit an infinite increase of the* $z_1^*$ *component is required to deteriorate* $z_2^*$. *Therefore, we deduce that no element* $c_+ \in C_+ = \{x \in \mathbb{R}^2 \mid x_i > 0, i = 1, 2\}$ *exists such that* $(1, 0)$ *minimises the problem* $\min_{z \in S} c_+^T z$ *and thus,* $(1, 0)$ *is not contained in the properly efficient set. Analogously we can deduce that* $(0, 1)$ *is also not a member of the set of properly efficient* *points.*

Since the functionals $\varphi(z)$ generated by the elements $c_+ \in C^+$ are strictly C-monotone by Remark 2.5 we can state the following important theorem.

**Theorem 2.11** *Let $S, C \in \mathbb{R}^k$ be sets. Then the following statements are true.*

- *If $C$ is a convex cone, then*

$$P(S, C) \subseteq E(S, C).$$

- *If $C$ is a closed, convex and pointed cone and the set $S$ is closed and convex then*

$$E(S, C) \subseteq cl(P(S, C)).$$

*Proof.* Concerning the first statement, suppose $\tilde{z} \notin E(S, C)$. If $\tilde{z} \notin S$ then it is apparent that $\tilde{z} \notin P(S, C)$. Therefore, assume that $\tilde{z} \in S \backslash E(S, C)$ and thus, deduce that there exists a $z \in S$ such that $\tilde{z} - z \in C \backslash \{0\}$. Consequently, we obtain by the definition of the dual cone that for all $c_+ \in C^+$

$$\begin{aligned} (c_+)^T(\tilde{z} - z) &> 0 \\ (c_+)^T\tilde{z} - (c_+)^T z &> 0 \\ (c_+)^T\tilde{z} &> (c_+)^T z. \end{aligned}$$

Hence, $\tilde{z} \notin P(S, C)$ which was to be shown.

Regarding the second proposition we refer the reader to Sawaragi et al. [39, pages 74–75]. □

So far we examined the necessary theory to prove that the set of properly efficient points is dense within the efficient set if certain prerequisites are satisfied. In order to take the next step and constitute the link between this result and the optimisation problem (2.1) we have to formulate the last theorem of this chapter first. The observation we present is that if we require the feasible set $\Omega$ to be convex, the objective function $f$ to be convex and employ the standard ordering cone $C = \mathbb{R}^k_+$ then the set $f(\Omega) + C$ is convex.

**Theorem 2.12** *Let $\Omega \subset \mathbb{R}^n$ be a convex set, $C = \mathbb{R}^k_+$ the standard ordering cone and $f$ a convex function $f : \mathbb{R}^n \to \mathbb{R}^k$. Then the set $f(\Omega) + C$ is convex.*

*Proof.* For $z^1, z^2 \in f(\Omega) + C$ there exist $x^1, x^2 \in \Omega$ and $c^1, c^2 \in C$ such that

$$z^i = f(x^i) + c^i$$

for $i = 1, 2$. Hence, we can derive for $\alpha \in [0, 1]$, $x^3 \in \Omega$ and $c^3, c^4, c^5 \in C$ that

$$\begin{aligned}
\alpha z^1 + (1-\alpha) z^2 &= \alpha(f(x^1) + c^1) + (1-\alpha)(f(x^2) + c^2) \\
&= \underbrace{\alpha f(x^1) + (1-\alpha) f(x^2)}_{\leqslant f(\alpha x^1 + (1-\alpha)x^2)\ (f\ convex)} + \underbrace{\alpha c^1 + (1-\alpha) c^2}_{=: c^3 \in C\ (C\ convex)} \\
&= f(\underbrace{\alpha x^1 + (1-\alpha) x^2}_{=: x^3 \in \Omega\ (\Omega\ convex)}) \qquad \underbrace{+c^4}_{\in C\ (C\ non-negative\ orthant)} \qquad +c^3 \\
&= f(x^3) \qquad \underbrace{+c^4 + c^3}_{=: c^5 \in C\ (Theorem\ 2.2)} \\
&\subset f(\Omega) + C.
\end{aligned}$$

□

With all these theoretical results at our deposal, we can establish the connection with the general multiobjective optimisation problem (2.1) under the following assumptions. We require the feasible set $\Omega \subset \mathbb{R}^n$ to be a compact convex set, the related objective function $f$ to be convex and have to utilise the standard order cone.

Initially, we want to point out that the second statement of Theorem 2.11 holds particularly for bounded, closed and convex sets $S$. If we consider problem (2.1) then the image $f(\Omega)$ forms exactly this set. The condition that $f(\Omega)$ is closed and bounded is satisfied due to the fact that the image $f(\Omega)$ of a continuous function $f : \Omega \to \mathbb{R}^k$ for a compact set $\Omega \subset \mathbb{R}^n$ is compact (i.e. closed and bounded) in the finite-dimensional space $\mathbb{R}^k$ [31, page 84–85]. Here we note that the continuity of $f$ is secured by Theorem 2.1. However, the convexity of $f(\Omega)$ is not guaranteed. Therefore, we exploit the result from Theorem 2.6 that $f(\Omega) + C$ possesses the same efficient set as $f(\Omega)$. In combination with Theorem 2.12, from which we obtain that $f(\Omega) + C$ is convex, we can determine a convex closed and bounded subset of $f(\Omega) + C$ that covers $f(\Omega)$. Hence, altogether we deduce that if we assume the prerequisites of Theorem 2.12 the set of properly efficient points is dense within the efficient set for the multiobjective optimisation problem (2.1).

Consequently, we conclude that we can determine properly efficient points by solving the following scalarised optimisation problem for $c_+ \in C^+$

$$\begin{aligned} \min \quad & c_+^T z \\ \text{s.t.} \quad & z \in f(\Omega). \end{aligned} \tag{2.6}$$

Furthermore, we stress that the standard ordering cone $C = \mathbb{R}^k_+$ is utilised in our context and therefore the dual cone is also the $\mathbb{R}^k_+$. Hence, we can interpret the objective functions of these ersatz problems (2.6) as weighted sums of the individual objective functions, where we denote the different weights by $\omega_i$ for all $i = 1, \ldots, k$. That is, we aim to minimise the function

$$c_+^T z = \sum_{i=1}^{k} \omega_i f_i(x)$$

with $\omega_i > 0$ for all $i = 1, \ldots, k$. In addition, without loss of generality we norm the sum of weights to 1. Altogether we can reformulate the optimisation problems we have to solve to determine properly efficient points as

$$\begin{aligned} \min \quad & \textstyle\sum_{i=1}^{k} \omega_i f_i(x) \\ \text{s.t.} \quad & x \in \Omega \end{aligned} \tag{2.7}$$

where $\omega_i > 0$ for all $i = 1, \ldots, k$ and $\sum_{i=1}^{k} \omega_i = 1$. Finally, computing sufficiently many solutions to problems of the form (2.7) for different weighting vectors $\omega = (\omega_1, \ldots, \omega_k)$ leads to good approximations of the set of efficient points, since all solutions combined are dense within the efficient set.

We proceed in this thesis by examining the solution method we employ to solve these scalarised ersatz problems (2.7) and investigating our technique to specify the different weights $\omega$ in the subsequent chapters.

# Chapter 3

# Interior-Point-Methods

In this chapter we investigate the subject area of interior-point methods. These techniques compute a sequence of iterates within the relative interior of a specific set, which eventually approaches a solution of an optimisation problem. As a starting point a general definition of a nonlinear optimisation problem is constituted and different types of solutions for this class of problems are specified. Subsequently necessary conditions for the optimality of solutions are reviewed. Furthermore, a numerical approach via a modified Newton method to determine the iterates mentioned above is established. As a next step the fundamental idea of the central path is examined. This is an arc of strictly feasible points that converges to a solution without moving towards the nonnegative boundary prematurely. The proof of the existence and uniqueness of the central path completes our analysis of this topic.

The following discussion of this subject focuses primarily on the aspects that are crucial for the development of our solution method. The research on our algorithm is motivated by the findings for linear programmes attained by Wright [48, Chapters 1-2] and the extensions to the convex quadratic case by Heermann [18, Chapter 3].

## 3.1 General nonlinear optimisation problem

A general optimisation problem can be stated as

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega. \end{aligned} \tag{3.1}$$

In this section we concentrate on problems which feature a twice-differentiable nonlinear objective function $f : \mathbb{R}^n \to \mathbb{R}$, a $n$-dimensional variable vector $x = (x_1, \dots, x_n) \in \mathbb{R}^n$ and a feasible set $\Omega$ being a subset of $\mathbb{R}^n$, i.e. $\Omega \subset \mathbb{R}^n$.

**Remark 3.1** *We want to point out that the objective function under consideration in this context is the weighted sum $\sum_{i=1}^{k} \omega_i f_i(x)$ of the single objective functions $f_i(x)$ as explained at the end of the previous chapter.*

**Remark 3.2** *Furthermore, we shall mention at this point that our approach in this thesis is to remain as general as possible with our assumptions, especially with regard to the nonlinearity property of the objective function $f$. However, we do assume convexity of the objective functions $f_1, ...f_k$, and consequently $f$, in the entirety of our following theoretical discussion. We explicitly highlight the critical points where this assumption proves to be crucial for our analysis.*

In order to specify solutions to problem (3.1) we introduce the following definitions of minimum points.

**Definition 3.1 (Local minimum point)** *Let $\|\cdot\|$ be the Euclidean norm*[1] *and consider the optimisation problem (3.1). A point $x^* \in \Omega$ is called a **local minimum point** of $f$ over $\Omega$ if $f(x^*) \leqslant f(x)$ for all $x \in \Omega$ satisfying $\|x - x^*\| \leqslant \varepsilon$ for some $\varepsilon > 0$.*

[1]The Euclidean norm $\|\cdot\|$ for a vector $x = (x_1, \dots, x_n) \in \mathbb{R}^n$ is defined as $\|x\| := \sqrt{(\sum_{i=1}^{n} |x_i|^2)}$

**Definition 3.2 (Global minimum point)** *Consider Problem (3.1). A point $x^* \in \Omega$ is said to be a **global minimum point** of $f$ over $\Omega$ if $f(x^*) \leqslant f(x)$ for all $x \in \Omega$.*

Hence, a local minimum point $x_l^*$ for Problem (3.1) is named as a local solution and a global minimum point $x_g^*$ for (3.1) is labelled a global solution.
As highlighted by several authors (compare for instance [29, page 168] or [1, page 4]) the search for solutions is often limited to local minimum points in practice. Firstly, this is due to theoretical assumptions since in general convexity properties are required to determine global solutions. And secondly, numerical calculations within iterative procedures are normally bounded to local search areas. Thus, for simplicity we refer to local solutions as optimal solutions in the subsequent analysis.

We proceed by scrutinising the existence of such minimum points. The theoretical concept that secures the existence of optimal solutions to problem (3.1) is Weierstrass' theorem which is presented next.

**Theorem 3.1 (Weierstrass' theorem)** *Let $\Omega \subset \mathbb{R}^n$ be a nonempty, compact subset of $\mathbb{R}^n$ and let $f : \Omega \to \mathbb{R}$ be a continuous function on $\Omega$. Then Problem (3.1) attains its minimum. That is, there exists a minimising solution to it.*

*Proof.* See [3, pages 41–42]. □

Therefore, with the assumption of a continuous objective function $f$ and a compact feasible set $\Omega$, the existence of an optimal solution for problem (3.1) is guaranteed by the above theorem.

## 3.2 Optimality criteria

Before we constitute necessary optimality criteria to generate optimal solutions to problem (3.1) we recall the definitions of the Jacobian matrix and the Hessian matrix and specify the feasible set in more details.

### 3.2.1 Preliminaries

**Definition 3.3 (Jacobian matrix)** *Let $f = (f_1, \ldots, f_k)$ be a differentiable function $f : \mathbb{R}^n \to \mathbb{R}^k$. The* ***Jacobian matrix*** *$\nabla f(x) \in \mathbb{R}^{k \times n}$ at the point $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is defined as*

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_k(x)}{\partial x_1} & \cdots & \frac{\partial f_k(x)}{\partial x_n} \end{pmatrix}.$$

**Definition 3.4 (Hessian matrix)** *Let $f$ be a twice-differentiable function $f : \mathbb{R}^n \to \mathbb{R}$. The* ***Hessian matrix*** *$\nabla^2 f(x) \in \mathbb{R}^{n \times n}$ at the point $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is defined as*

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f(x)}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_n \partial x_n} \end{pmatrix}.$$

In addition, we specify the set $\Omega$ by introducing inequality and equality constraints. Thus, we redefine the feasible set as

$$\Omega := \{x \in \mathbb{R}^n | \; g(x) \leqslant 0, h(x) = 0\},$$

where $g(x) = (g_1(x), \ldots, g_p(x))$ and $h(x) = (h_{p+1}(x), \ldots, h_{p+q}(x))$ are continuously differentiable functions $g : \mathbb{R}^n \to \mathbb{R}^p$ and $h : \mathbb{R}^n \to \mathbb{R}^q$. Consequently, we consider the following modified optimisation problem in the subsequent examination

$$\begin{array}{lrcl} \min & f(x) & & \\ \text{s.t.} & g(x) & \leqslant & 0 \\ & h(x) & = & 0. \end{array} \tag{3.2}$$

### 3.2.2 KKT - conditions

The famous Karush-Kuhn-Tucker conditions (KKT-conditions) which allow us to identify local solutions rely on two important concepts which we have to introduce first. Initially, we define the active set and subsequently formulate the linear independence constraint qualification.

**Definition 3.5 (Active set)** *The **active set** with respect to $x$ is defined as*

$$\mathcal{A}(x) := \{i \in \{1, \ldots, p\} \mid g_i(x) = 0\} \cup \{j = p+1, \ldots, p+q \mid h_j(x) = 0\}.$$

*That is, it represents the union of all indices of the equality constraints and these indices of the inequality constraints that are active (i.e. $g_i(x) = 0$ for $i \in \{1, \ldots, p\}$).*

In the following we utilise the version of the KKT-conditions stated in the book by Nocedal and Wright [35, page 328]. These optimality conditions employ the idea of a linear independence constraint qualification (LICQ) which we establish next.

**Definition 3.6 (LICQ)** *Let $x^* \in \Omega$ be a point and the active set $\mathcal{A}(x^*)$ as defined above. The **linear independence constraint qualification** (LICQ) holds at point $x^*$ if the set of active constraint gradients $\{\nabla g_i(x^*) \mid i \in \mathcal{A}(x^*)\} \cup \{\nabla h_j(x^*) \mid j \in \mathcal{A}(x^*)\}$ is linearly independent.*

Now we are able to state the KKT-conditions which provide us with necessary optimality conditions to determine optimal solutions.

**Theorem 3.2 (KKT - conditions)** *Let $x^*$ be a local solution for the optimisation problem (3.2) and assume that the LICQ is satisfied at $x^*$. Then there exist so-called Lagrange* *multiplier vectors $\lambda \in \mathbb{R}^q$ and $s \in \mathbb{R}^p$ such that*

$$\begin{aligned}
\nabla f(x^*)^T - \nabla h(x^*)^T \lambda + \nabla g(x^*)^T s &= 0 \\
g(x^*) &\leqslant 0 \\
h(x^*) &= 0 \\
s &\geqslant 0.
\end{aligned}$$

*Furthermore, the following complementarity conditions are satisfied*

$$s_i g_i(x^*) = 0 \quad \text{for all} \quad i = 1, \ldots, p.$$

*Proof.* Compare [35, pages 341–342]. □

With this major result at hand we are equipped with the theoretical foundation to design a method to compute solutions for nonlinear problems of the form (3.2). The details of this technique are illustrated in the following sections.

## 3.3 Particular nonlinear optimisation problems

Up to this point we presented optimisation problems in the most general formulation that was appropriate. However, we now focus on problems with more specific properties in order to avoid unnecessary technical complexity. More precisely, we are investigating nonlinear optimisation problems, which can be regarded as special cases of the optimisation problem (3.2). In particular we specify the feasible set by defining the inequality contraints as $g(x) = -x$ and the equality constraints as $h(x) = Ax - b$, respectively. Here, $A$ is assumed to be a matrix $A \in \mathbb{R}^{n \times m}$ with full row rank and $m < n$, whereas $b$ is a $m$-dimensional vector, i.e. $b \in \mathbb{R}^m$. This formulation of the constraints is widely known as standard form.

Based on these specifications we point out that the nonlinear problems under consideration in the subsequent discussion are of the particular form

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & Ax = b \\ & x \geqslant 0. \end{aligned} \tag{3.3}$$

We proceed by applying the KKT-conditions to optimisation problem (3.3) and result in the following equalities and inequalities

$$\begin{aligned} \nabla f(x)^T - A^T\lambda \ - s &= 0 \\ -x &\leqslant 0 \\ Ax - b &= 0 \\ s &\geqslant 0 \\ -s_i x_i &= 0 \quad \text{for all} \quad i = 1, \ldots, n \end{aligned}$$

where $\lambda \in \mathbb{R}^m$, $s \in \mathbb{R}^n$ and $I_n$ denotes the identity matrix of the $\mathbb{R}^n$. We note that this system needs to be solved to find an optimal solution for (3.3). Moreover, for our upcoming analysis it proves to be advantageous to rewrite the KKT-conditions in a more compact way as

$$F(w) := \begin{bmatrix} -\nabla f(x)^T + A^T\lambda + s \\ Ax - b \\ XSe \end{bmatrix} = 0 \tag{3.4}$$
$$(x, s) \geqslant 0$$

where $w := (x, \lambda, s)$ is a shorthand notation, $X = \text{diag}(x_1, \ldots, x_n)$, $S = \text{diag}(s_1, \ldots, s_n)$, $e = (1, \ldots, 1)^T \in \mathbb{R}^n$ and $(x, s)$ is a shorthand notation for the vector $(x^T, s^T)^T$, where the inequality has to hold element-wise, i.e. $x_i \geqslant 0$ for all $i = 1, \ldots, n$ and $s_i \geqslant 0$ for all $i = 1, \ldots, n$. Ultimately, we introduce a notation that is used later on in our discussion.

**Definition 3.7** *The **feasible set** for system (3.4) is defined as*

$$\mathcal{F} := \{(x, \lambda, s) \in \mathbb{R}^{n \times m \times n} \mid -\nabla f(x) + A^T \lambda + s = 0, Ax = b, (x, s) \geqslant 0\}.$$

*and the **strictly feasible set** for system (3.4) is defined as*

$$\mathcal{F}^0 := \{(x, \lambda, s) \in \mathbb{R}^{n \times m \times n} \mid -\nabla f(x) + A^T \lambda + s = 0, Ax = b, (x, s) > 0\}.$$

*The definition of $\mathcal{F}^0$ merely implies that this set incorporates triples that satisfy the first two equations of the KKT-conditions and requires the vectors $x$ and $s$ to be strictly positive.*

In the context of linear programming Nocedal and Wright [35, page 397] use analogous definitions and due to the structure of the dual problem to the linear optimisation problem name them primal-dual feasible and primal-dual strictly infeasible set.

## 3.4 Primal-dual interior-point methods

In order to compute solutions to system (3.4) (i.e. find a $w^*$ such that $F(w^*) = 0$) we apply a modified Newton method. That is, starting for an initial point $w^0 := (x^0, \lambda^0, s^0)$ we compute a series of iterates according to the rule

$$w_{k+1} = w_k + \Delta w_k = w_k - (\nabla F(w_k))^{-1} F(w_k)$$

for all $k \geqslant 0$ which leads to a solution, where $\nabla F(w)$ denotes the Jacobian matrix of the function $F(w)$. Therefore, assuming the non-singularity of the Jacobian matrix, we want to compute a solution vector $\Delta w := (\Delta x, \Delta \lambda, \Delta s)$ for the system

$$\nabla F(w) \Delta w = -F(w)$$

for a given $w := (x, \lambda, s)$.

Hence, we have to examine the following system of equations

$$\begin{bmatrix} -\nabla^2 f(x) & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -r_{xs} \end{bmatrix} \tag{3.5}$$

where $\nabla^2 f(x)$ is the Hessian matrix of the function $f$, $r_c := -\nabla f(x)^T + A^T\lambda + s$, $r_b := Ax - b$ and $r_{xs} := XSe$.

Here we want to highlight that we prove the solvability of system (3.5) as well as determining the actual solution vector $\Delta w := (\Delta x, \Delta\lambda, \Delta s)$ in the following examination.

**Remark 3.3** *For easier legibility we denote the Hessian matrix $\nabla^2 f(x)$ by $H$ in the subsequent analysis.*
*Furthermore, we revert to several fundamental results of mathematics, a summary of which is offered in Appendix A.*

To secure that the system (3.5) is solvable we have to demand a certain property from the objective function $f$. We now derive which assumption proves to be sufficient.
If $(x, s) > 0$ is satisfied then the matrix $D := S^{-1/2}X^{1/2}$, where $S^{-1/2} := \operatorname{diag}(s_1^{-1/2}, \ldots, s_n^{-1/2})$ and $X := \operatorname{diag}(x_1^{1/2}, \ldots, x_n^{1/2})$, is positive definite and furthermore, the matrix $D^{-2} = SX^{-1}$ is also positive definite. If we now require $f$ to be convex then by Theorem A.1 the Hessian matrix $H$ is positive semi-definite. Subsequently, by Theorem A.2 the term $(D^{-2} + H)$ is positive definite and as a consequence of Theorem A.3 it is also nonsingular. Now since

$$(D^{-2} + H)^{-1} = ((I_n + HD^2)D^{-2})^{-1}$$

we can conclude by Corollary A.1 that $I_n + HD^2$ is nonsingular and thus $B := (I_n + HD^2)^{-1}$ exists. Moreover, since we assume that the matrix $A$ has full row rank, we derive from Theorem A.6 and Theorem A.7 that the term

$$AD^2BA^T = AD^2(I_n + HD^2)^{-1}A^T = A((I_n + HD^2)D^{-2})^{-1}A^T = A(D^{-2} + H)^{-1}A^T$$

is invertible.

With these intermediate results at hand we can scrutinise the system (3.5) in more details. The first step is to transform it to the following system of equations

$$\begin{array}{lllllll} -H\Delta x & + & A^T\Delta\lambda & + & \Delta s & = & -r_c \\ A\Delta x & & & & & = & -r_b \\ S\Delta x & & & + & X\Delta s & = & -r_{xs}. \end{array} \tag{3.6}$$

From the third equation of (3.6) we derive that

$$\begin{aligned} S\Delta x &= -r_{xs} - X\Delta s \\ \Delta x &= S^{-1}(-r_{xs} - X\Delta s) \\ &= -(S^{-1}r_{xs} + D^2\Delta s) \end{aligned} \tag{3.7}$$

where $D$ is defined as above.

Applying the result (3.7) to the first equation of (3.6) it follows that

$$\begin{aligned} H(S^{-1}r_{xs} + D^2\Delta s) + A^T\Delta\lambda + \Delta s &= -r_c \\ (HD^2 + I_n)\Delta s &= -r_c - HS^{-1}r_{xs} - A^T\Delta\lambda \\ \Delta s &= -B(r_c + HS^{-1}r_{xs} + A^T\Delta\lambda) \end{aligned} \tag{3.8}$$

where $B$ is defined as above.

Finally using both results (3.7) and (3.8) we can examine the second equation of (3.6) and show that

$$\begin{aligned} -A(S^{-1}r_{xs} + D^2\Delta s) &= -r_b \\ -AS^{-1}r_{xs} - AD^2(-B(r_c + HS^{-1}r_{xs} + A^T\Delta\lambda)) &= -r_b, \end{aligned}$$

which leads to

$$\begin{aligned} (AD^2BA^T)\Delta\lambda &= -r_b + AS^{-1}r_{xs} - AD^2B(r_c + HS^{-1}r_{xs}) \\ (AD^2BA^T)\Delta\lambda &= -r_b + A(S^{-1}r_{xs} - D^2B(r_c + HS^{-1}r_{xs})). \end{aligned} \tag{3.9}$$

Altogether we deduce that (3.5) can be transformed to the following system of equations which is equivalent to the original one.

$$\begin{aligned} (AD^2BA^T)\Delta\lambda &= -r_b + A(S^{-1}r_{xs} - D^2B(r_c + HS^{-1}r_{xs})), \\ \Delta s &= -B(r_c + HS^{-1}r_{xs} + A^T\Delta\lambda), \\ \Delta x &= -(S^{-1}r_{xs} + D^2\Delta s). \end{aligned} \tag{3.10}$$

With the explanations regarding the invertibility of the matrices $B$ and $AD^2BA^T$ presented above we conclude that system (3.10) is solvable and therefore the non-singularity of the Jacobian matrix follows. We re-emphasise that the assumption that the objective function $f$ is convex guarantees the theoretical solvability.

**Remark 3.4** *In the following we encounter several systems of linear equations that are similar to (3.5). These systems only differ in the right-hand side of the equations. The proof that these systems can be solved is analogous to the one just detailed.*

After one iteration of the Newton method is performed a new iterate $\tilde{w} = w + \Delta w$ is obtained. However, this $\tilde{w}$ often violates the non-negativity constraints $(x, s) \geqslant 0$. To avoid this effect we introduce a parameter $\alpha$ that adjusts the step length along the Newton direction. That is, we compute an $\alpha \in (0, 1]$ such that the new iterate

$$\tilde{w} = w + \alpha \Delta w$$

meets the nonnegativity conditions for $\tilde{x} = x + \alpha \Delta x$ and $\tilde{s} = s + \alpha \Delta s$.

In addition, the interior point algorithm that we are constructing possesses another vital property. Besides regulating the step length this method secures that individual components of $(x, s)$ do not approach the nonnegative boundary too quickly. Essential for this strategy is the central path which we present below. This path consists of strictly feasible points for which the pairwise products $x_i s_i$ for all indices $i = 1, \dots, n$ are identical in every iteration step.

**Remark 3.5** *The algorithm established in this thesis employs iterates that solely have to meet the positivity condition $(x, s) > 0$. However, the residues $r_b$ and $r_c$ do not need to be equal to zero and thus, the iterates are not necessarily elements of the strictly feasible set. Such points are labelled as infeasible and methods utilising such points are called infeasible interior-point algorithms. The motivation for this approach is that it usually proves to be exceptionally difficult to determine feasible starting points. Thus, a solution algorithm based on these assumptions must reduce not only $XSe$ to zero but also the residues $r_b$ and $r_c$. This procedure results in iterates that approach a solution of the feasible set $\Omega$. The process how this reduction of the residues is achieved in detail is illustrated in the next chapter.*

## 3.5 Central path

The equality of the products $x_i s_i$ for all indices $i = 1, \dots, n$ is achieved by introducing a parameter $\tau > 0$ to the system (3.4) in the following way

$$\begin{bmatrix} -\nabla f(x) + s + A^T\lambda \\ Ax - b \\ XSe \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \tau e \end{bmatrix} \tag{3.11}$$

$$(x, s) > 0$$

where $X = \operatorname{diag}(x_1, \dots, x_n)$, $S = \operatorname{diag}(s_1, \dots, s_n)$ and $e = (1, \dots, 1)^T \in \mathbb{R}^n$.

**Definition 3.8 (Central path)** *The **central path** is defined as*

$$\mathcal{C} := \{(x, \lambda, s) \in \mathcal{F}^0 \mid x_i s_i = \tau,\ \forall i = 1, \dots, n,\ \tau > 0\}$$

*and essentially aggregates all solutions to system (3.11) for arbitrary $\tau > 0$.*

We note that the central path constitutes a feasible path in a sense that the KKT-condition

$$-\nabla f(x) + s + A^T\lambda = 0$$

and the side constraints

$$Ax = b$$

are ”strictly” satisfying the equalities. Contrary, infeasible paths do not have to satisfy the these equalities and hence, residuals $r_b \neq 0$ and $r_c \neq 0$ are introduced in the following way

$$-\nabla f(x) + s + A^T\lambda = r_c$$

and

$$Ax - b = r_b.$$

Concerning the existence and the uniqueness of the central path we introduce the following logarithmic barrier function

$$f_\tau(x) := f(x) - \tau \sum_{i=1}^{n} \log(x_i)$$

and consider the following related problem

$$\begin{array}{lrcl} \min & f_\tau(x) & & \\ \text{s.t.} & Ax & = & b \\ & x & > & 0. \end{array} \qquad (3.12)$$

In the subsequent discussion we determine a property of $f$ that leads to an unique solution for the logarithmic barrier function problem. Furthermore, we show that (3.11) and (3.12) have uniquely corresponding solutions. Finally, we complete our analysis with a proof of the existence of a solution for system (3.12) employing the concept of level sets.

From a well-known theorem of optimisation (compare for instance [48, page 242-243]) we know that if an objective function is strictly convex on its feasible region any local solution of this function is an uniquely defined global solution.
In order to achieve strict convexity of the objective function of problem (3.12) we must examine all terms of this function and detect which property the function $f$ has to feature for $f_\tau$ to become strictly convex.

**Theorem 3.3** *If the original objective function $f$ is convex then the logarithmic barrier function $f_\tau$ is strictly convex.*

*Proof.* The summation term of the logarithmic barrier function (i.e. $-\tau \sum_{i=1}^{n} \log(x_i)$) has a positive definite Hessian matrix for all $x \neq 0$ and for all $\tau > 0$. Thus, by Theorem A.1 it is strictly convex on the feasible set for problem (3.12). Hence, if we assume the original objective function $f$ to be convex we obtain by Theorems A.1 and A.2 that the logarithmic barrier function is strictly convex. □

We exploit this interrelation to prove a theorem regarding the solution sets of problems (3.11) and (3.12).

**Theorem 3.4** *Let $\tau > 0$, $f$ be a convex function and $A$ a matrix with full row rank. Furthermore, assume that the strictly feasible set is nonempty, i.e. $\mathcal{F}^0 \neq \emptyset$. Then the following statements are equivalent:*

- *the central path system (3.11) has a unique solution $(x^*, \lambda^*, s^*)$.*
- *the logarithmic barrier function problem (3.12) has a unique solution $x_\tau$.*

*Proof.* We begin by showing that the first statement can be derived from the latter one. If we apply the KKT-conditions to problem (3.12) we derive that the unique solution $x_\tau$ must necessarily satisfy the following system of equations

$$\begin{aligned} \nabla f(x_\tau) - \tau X_\tau^{-1} e - A^T \lambda_\tau &= 0 \\ A x_\tau &= b \\ x_\tau &> 0. \end{aligned} \tag{3.13}$$

If we set $x^* := x_\tau$ (unique), $s^* := \tau X_\tau^{-1} e > 0$ and $\lambda^* := \lambda_\tau$, it follows that $(x^*, \lambda^*, s^*)$ is a solution for system (3.11) which defines the central path. By the definition of $s^*$ we conclude that it is uniquely determined. Moreover, with the assumption that $A$ features full row rank we derive from the equation $-\nabla f(x^*) + s^* + A^T \lambda^* = 0$ that $\lambda^*$ can also be uniquely identified. Hence, we have shown that a unique solution for (3.11) exists if there is a unique solution for the logarithmic barrier function problem.

Conversely, we now prove that the first condition implies the second one. Therefore, consider a unique solution $(x^*, \lambda^*, s^*)$ for system (3.11) and define $x_\tau := x^*$ and $\lambda_\tau := \lambda^*$. Furthermore, deduce from the third equation of (3.11) that $s^* = S^* e = \tau X^{*-1} e = \tau X_\tau^{-1} e$. We observe that this solution $(x^*, \lambda^*, s^*)$ also meets the KKT-conditions (3.13) and thus, solves the logarithmic barrier function problem. Lastly, since the objective function for this problem (3.12) is strictly convex by Theorem 3.3 we conclude with the well-known theorem of optimisation outlined before that this solution is unique. □

Hitherto one crucial prerequisite has been generously assumed, namely the existence of a solution for problem (3.12). Hence, it remains to show that such an unique solution exists. For this proof we use a common technique exploiting the fact that level sets, defined as

$$\mathcal{L}_\tau(\kappa) := \{x \in \mathbb{R}^n \mid \exists \lambda, s : (x, \lambda, s) \in \mathcal{F}^0, f_\tau(x) \leqslant \kappa\},$$

are compact for all $\kappa > 0$ under certain assumptions. More precisely, we have to require that the logarithmic barrier function diverges to infinity for $x_i \longrightarrow 0$ as well as for $x_i \longrightarrow \infty$ for all $i = 1, \ldots, n$. In other words, $f(x)$ has to grow faster to $+\infty$ than $\tau \sum_{i=1}^n \log(x_i)$ for $x_i \longrightarrow \infty$ for all $i = 1, \ldots, n$. Analogously, $\tau \sum_{i=1}^n \log(x_i)$ has to decrease faster to $-\infty$ than $f(x)$ for $x_i \longrightarrow 0$ for all $i = 1, \ldots, n$.

**Theorem 3.5** *Let $\tau > 0$ be fixed and let $f$ be a function such that $f(x) - \tau \sum_{i=1}^n \log(x_i) \longrightarrow +\infty$ for $x_i \longrightarrow 0$ and $x_i \longrightarrow +\infty$ for all $i = 1, \ldots, n$. Then the level sets $\mathcal{L}_\tau(\kappa)$ are compact for all $\kappa > 0$.*

*Proof.* We see from the definition of $\mathcal{L}_\tau(\kappa)$ in combination with our required divergence properties for $x_i \longrightarrow 0$ for all $i = 1, \ldots, n$ that the level sets are closed. In combination with the requested divergence properties for $x_i \longrightarrow +\infty$ we derive that the logarithmic barrier function $f_\tau(x) := f(x) - \tau \sum_{i=1}^{n} \log(x_i) \longrightarrow +\infty$ for all $i = 1, \ldots, n$. However, since we established an upper bound $\kappa$ for $f_\tau(x),\ x \in \mathcal{L}_\tau(\kappa)$, we conclude the level sets are bounded. $\square$

Assuming the preconditions of the above theorems and the non-emptiness of $\mathcal{F}^0$ we can apply Weierstrass' theorem to the continuous function $f_\tau$ and a compact level set $\mathcal{L}_\tau(\kappa)$ to derive that a minimum for the problem

$$\begin{array}{ll} \min & f_\tau(x) \\ \text{s.t.} & x \in \mathcal{L}_\tau(\kappa) \end{array} \tag{3.14}$$

exists for a $\kappa > 0$. Altogether, we can formulate the theorem that secures the uniqueness of the central path next.

**Theorem 3.6** *Suppose all assumptions of the previous theorems are satisfied. Then for every $\tau > 0$ the system (3.11) has a unique solution.*

*Proof.* Consider an arbitrary tripel $(\tilde{x}, \tilde{\lambda}, \tilde{s}) \in \mathcal{F}^0$. We can construct the level set $\mathcal{L}_\tau(\kappa)$ by defining $\kappa$ as $\kappa := f_\tau(\tilde{x})$. From Theorem 3.5 we know that $\mathcal{L}_\tau(\kappa)$ is compact. Furthermore, it follows the explanations above that a solution to the logarithmic barrier problem (3.12) exists and is contained in this level set. Finally, exploiting Theorem 3.4 we conclude that a unique solution for the central path system (3.11) exists. $\square$

Ultimately, we complete our investigation of interior-point methods by presenting the final theorem that states that the central path converges towards an optimal solution of the optimisation problem (3.3).

**Theorem 3.7** *For an algorithm computing triples $(x^k, \lambda^k, s^k)$ for $\tau_k > 0$ and $\tau_{k+1} < \tau_k$ for all $k = 1, 2, \ldots$, the corresponding central path converges to a unique solution $(x^*, \lambda^*, s^*)$ as $\tau_k \longrightarrow 0$.*

*Proof.* We refer the reader to [34]. $\square$

# Chapter 4

# Infeasible-Path-Following Algorithm

In this chapter we establish the infeasible-path-following algorithm NLIPF which determines solutions for nonlinear optimisation problems with linear constraints. At first we define and analyse a specific neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$ of the central path to which all iterates of the solution method are restricted. This is followed by a motivation of our technique and a structured formulation of the procedures in details. We proceed by introducing useful modifications to the residuals and consequently deducing relationships between them which prove to be crucial for our further examination. Subsequently we state assumptions which have to hold to be in a position to show that the algorithm approaches a solution. Finally we conclude this chapter by presenting the proof of convergence which comprises of a few technical results in the beginning followed by the main result which relies on positive step lengths in each iteration of the Newton method.

## 4.1 Neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$

As a starting point we briefly recapitulate the nonlinear optimisation problem under consideration

$$\begin{array}{lrcl} \min & f(x) & & \\ \text{s.t.} & Ax & = & b \\ & x & \geqslant & 0 \end{array}$$

and restate the residuals for an arbitrary triple $(x, \lambda, s)$

$$r_b = Ax - b \quad \text{and} \quad r_c = -\nabla f(x) + s + A^T\lambda.$$

Furthermore, we introduce the subsequent variable which is frequently used in the up-coming investigation

$$\mu := \frac{x^T s}{n}.$$

In the context of linear programming the variable $\mu$ defines the 'duality gap' between the objective values of the primal optimisation problem and its dual problem. Consequently, we exploit the concept analogously and adapt it to the nonlinear case.
For a particular iterate of the algorithm $(x^k, \lambda^k, s^k)$ we denote these vectors by $r_b^k$ and $r_c^k$, respectively and specify the term $\mu$ as $\mu_k := ((x^k)^T s^k)/n$.

Following these preliminary remarks we constitute the neighbourhood of the central path which we utilise in our solution method and discuss it's fundamental properties afterwards.

**Definition 4.1** *Let $\gamma \in (0, 1)$, $\beta \geqslant 1$ be given parameters and let $\mu_0$ and $(r_b^0, r_c^0)$ be the terms evaluated at the starting point of the algorithm $(x^0, \lambda^0, s^0)$. We define the* *neighbourhood of the central path as*

$$\mathcal{N}_{-\infty}(\gamma,\beta) := \{(x,\lambda,s) \mid \|(r_b,r_c)\| \leqslant \frac{\|(r_b^0,r_c^0)\|}{\mu_0}\beta\mu, (x,s) > 0, x_is_i \geqslant \gamma\mu, i = 1,\ldots,n\}$$

*where* $\|\cdot\|$ *represents the Euclidean norm.*

The subscript $-\infty$ relates to a 'minimum norm' (the converse of a maximum norm) since all elements of $\mathcal{N}_{-\infty}(\gamma,\beta)$ have to satisfy the condition $x_is_i \geqslant \gamma\mu$ for all $i = 1,\ldots,n$ and hence $\min_i x_is_i \geqslant \gamma\mu$. Furthermore, we want to highlight that the condition $\beta \geqslant 1$ guarantees that the starting point $(x^0,\lambda^0,s^0)$ is contained in the set $\mathcal{N}_{-\infty}(\gamma,\beta)$. In addition, one essential feature of the technique we are designing in this chapter is that all iterates $(x^k,\lambda^k,s^k)$ also have to belong to $\mathcal{N}_{-\infty}(\gamma,\beta)$. Furthermore, we observe that the infeasibility with respect to the residuals $r_b^k$ and $r_c^k$ of every triple $(x^k,\lambda^k,s^k)$ restricted to the set $\mathcal{N}_{-\infty}(\gamma,\beta)$ is bounded by a multiple of the parameter $\mu_k$. Thus, an immediate consequence from the necessity that all iterates must be included in $\mathcal{N}_{-\infty}(\gamma,\beta)$ is that if we reduce $\mu_k$ to 0 the residuals $(r_b^k,r_c^k)$ will also converge to zero, i.e. $r_b^k \to 0$ and $r_c^k \to 0$ for $k \to \infty$. Lastly, the conditions $x_is_i \geqslant \gamma\mu$ ensure that individual components of $(x,s)$ do not approach zero at a premature stage and hence, prevent distorted search directions.

## 4.2 Algorithm NLIPF

We enhance the algorithm IPF suggested by Wright [48, Chapter 6] to nonlinear objective functions and name it NLIPF, which stands for Nonlinear Infeasible-Path-Following Algorithm. Nocedal and Wright [35, page 401 et seqq.] studied this approach for the linear case, while Wang and Fei [46] exploit a different neighbourhood in their algorithm for convex programming. Moreover, we provide the following shorthand notation which we employ in the subsequent structuring of the modus operandi

$$(x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k)) := (x^k, \lambda^k, s^k) + \alpha_k(\Delta x^k, \Delta\lambda^k, \Delta s^k),$$

and

$$\mu_k(\alpha_k) := \frac{(x^k(\alpha_k))^T s^k(\alpha_k)}{n}.$$

Finally, we want to mention that the basis of our algorithm is the application of Newton's method to the central path system (3.11) with the incorporation of the residuals $r_b$, $r_c$ and $r_{xs}$.

**Algorithm NLIPF**

- Initialisation
  - given $\varepsilon, \gamma, \beta, \sigma_{min}, \sigma_{max}$ with $\varepsilon > 0, \gamma \in (0,1), \beta \geqslant 1$ and $0 < \sigma_{min} < \sigma_{max} < 0.5$
  - choose a starting point $(x^0, \lambda^0, s^0)$ with $(x^0, s^0) > 0$
  - set $k := 0$
  - compute $\mu_0 := ((x^0)^T s^0)/n$
- Main loop

  while $\mu_k > \varepsilon$

  - choose $\sigma_k \in [\sigma_{min}, \sigma_{max}]$ and solve

$$\begin{bmatrix} -\nabla^2 f(x^k) & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta \lambda^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} -r_c^k \\ -r_b^k \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix} \tag{4.1}$$

  - choose $\alpha_k$ as the largest value of $\alpha \in [0,1]$ such that

$$(x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k)) \in \mathcal{N}_{-\infty}(\gamma, \beta)$$

    and the following sufficient decrease condition holds

$$\mu_k(\alpha_k) \leqslant (1 - 0.01\alpha_k)\mu_k$$

```
    – set
        $(x^{k+1}, \lambda^{k+1}, s^{k+1}) := (x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k)),$
        $k := k+1$
        $\mu_k := ((x^{k+1})^T s^{k+1})/n$
end (while)
```

**Remark 4.1** *The method we create in the thesis pursues a strategy stemming from linear programming and utilises the maximum value for the step length parameter $\alpha$. This route is chosen for sake of simplicity, however other approaches have been mentioned in the literature, e.g. [48, page 110], and particulary in the context of nonlinear programming new filter methods have been discussed in recent papers, e.g. [44] and [45].*

The introduction of the centering parameters $\sigma_k$ in system (4.1) serves the purpose of establishing a balance between two different goals. Obviously one crucial aim of the algorithm is to generate significant reduction in $\mu$. This is achieved by setting $\sigma_k$ close to zero, in which case standard Newton steps are computed. However, the iterates rapidly move towards the boundary of the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$ if $\mu$ is decreased considerably. Therefore, a Newton step along a centering direction proves to be beneficial. Choosing larger values for $\sigma_k$ leads to iterates that are near the central path. These well-centred points promise remarkable progress with regard to $\mu$ in the next iteration.
Wright [48, chapter 5] outlines several different strategies to reach both goals. For instance, predictor-corrector methods compute one step with $\sigma_k = 0$ (predictor step) followed by one iteration with $\sigma_{k+1} = 1$ (corrector step) and hence, take maximum advantage of both approaches. Another method is to determine $\sigma_k$ depending on the reduction of $\mu_{k-1}$ in the previous iteration. The advantage of specifying the centering parameter in real time is that the most suitable strategy is selected automatically.

## 4.3 Residuals $r_b^k$ and $r_c^k$

Since the residuals $r_b^k$ and $r_c^k$ have to be decreased to zero in the course of the algorithm we need to analyse the relationships between them for different $k$ in order to establish convergence results. At the outset of this section we define a scalar quantity $\nu_k$ which proves to be extremely valuable in the following examination. Let $\nu_0 := 1$ and

$$\nu_k := \prod_{j=0}^{k-1}(1-\alpha_j)$$

where $\alpha_j$ are the step length parameters of the algorithm NLIPF for $j \geqslant 0$. It follows immediately from these definitions that $0 \leqslant \nu_k \leqslant 1$ for all $k \geqslant 0$.

The next theorem points out how the residuals $r_b^k$ and $r_c^k$ are related to each other. In order to achieve this desirable property we have to modify the residuals $r_c^k$ by introducing error terms $E_k$ for all $k \geqslant 0$ which provide for the inexactness approach that we are employing. Before we can analyse the residuals and modifications in more details we introduce a short-hand notation that is based on Taylor expansion in the following subsection.

### 4.3.1 Taylor Expansion

In this section key definitions of the Taylor expansion are summarised and a useful short-hand notation is established.

**Definition 4.2 (Taylor expansion)** *Let $f$ be an infinitely differentiable function $f : \mathbb{R}^n \to \mathbb{R}$. Then the **Taylor expansion** of the function $f$ at $x$ is given by*

$$f(x + \Delta x) = f(x) + \nabla f(x)\Delta x + T(x, \Delta x),$$

*where $T(x, \Delta x)$ is a shorthand notation for the remaining Taylor expansion terms.*

**Definition 4.3 (Multidimensional Taylor expansion)** *Let $f$ be an infinitely differentiable function $f : \mathbb{R}^n \to \mathbb{R}^n$. We define the* ***Taylor expansion*** *of the function $f$ at $x$ in the following fashion. We constitute a Taylor expansion for each individual component $f_i(x)$ $(i = 1, \dots, n)$ and combine the single remaining Taylor expansion terms to a Taylor expansion vector $\mathcal{T}$. That is,*

$$\begin{aligned} f(x + \Delta x) &= \begin{bmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{bmatrix} + \begin{bmatrix} \nabla_1 f_1(x) & \dots & \nabla_n f_1(x) \\ \vdots & \ddots & \vdots \\ \nabla_1 f_n(x) & \dots & \nabla_n f_n(x) \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{bmatrix} + \begin{bmatrix} T_1(x, \Delta x) \\ \vdots \\ T_n(x, \Delta x) \end{bmatrix} \\ &= f(x) + \nabla f(x)^T \Delta x + \mathcal{T}(x, \Delta x). \end{aligned}$$

### 4.3.2 Residuals $r_b^k$ and $r_c^k$

With the above notation at hand we redefine the residuals $r_c^k$ as

$$r_c^k = -\nabla f(x^k) + s^k + A^T \lambda^k + E_k$$

where $E_k := \sum_{i=0}^k \mathcal{T}_i$. Here, we utilised the shorthand notation

$$\mathcal{T}_0 := 0 \quad \text{and} \quad \mathcal{T}_i := \mathcal{T}(x^{i-1}, \alpha_{i-1} \Delta x^{i-1})$$

for the Taylor expansion vectors of the gradient functions $\nabla f(x^{i-1} + \alpha_{i-1} \Delta x^{i-1})$ for $i \geqslant 1$.

**Theorem 4.1** *For all $k \geqslant 0$ the residuals $r_b^k$ and $r_c^k$ satisfy the following equations*

$$r_b^k = \nu_k r_b^0 \quad \text{and} \quad r_c^k = \nu_k r_c^0. \tag{4.2}$$

*Furthermore, if the starting point* $(x^0, \lambda^0, s^0)$ *is infeasible, i.e.* $(r_b^0, r_c^0) \neq 0$*, then the following estimate also holds*

$$\nu_k \leqslant \frac{\beta}{\mu_0}\mu_k. \tag{4.3}$$

*Proof.* By the definition of the iterates $x^k$ we derive that

$$Ax^k - b = A(x^{k-1} + \alpha_{k-1}\Delta x^{k-1}) - b = Ax^{k-1} - b + \alpha_{k-1}A\Delta x^{k-1}.$$

Consequently, we conclude from this result, the definition of the residuals $r_b^k$ and the second row of (4.1) that

$$r_b^k = Ax^k - b = Ax^{k-1} - b + \alpha_{k-1}A\Delta x^{k-1} = r_b^{k-1} - \alpha_{k-1}r_b^{k-1} = (1 - \alpha_{k-1})r_b^{k-1}.$$

It follows by induction that

$$r_b^k = (1 - \alpha_{k-1})r_b^{k-1} = \prod_{j=0}^{k-1}(1 - \alpha_j)r_b^0 = \nu_k r_b^0.$$

Similarly to the case for the residuals $r_b^k$ we deduce with the revised definition of the residuals $r_c^k$, the definition of the iterates $(x^k, \lambda^k, s^k)$, the Taylor expansion analysis and the first row of (4.1) that

$$\begin{aligned}
r_c^k &= -\nabla f(x^k) + s^k + A^T\lambda^k + E_k \\
&= -\nabla f(x^{k-1} + \alpha_{k-1}\Delta x^{k-1}) + E_k \\
&\quad + s^{k-1} + \alpha_{k-1}\Delta s^{k-1} + A^T\lambda^{k-1} + \alpha_{k-1}A^T\Delta\lambda^{k-1} \\
&= -\nabla f(x^{k-1}) - \alpha_{k-1}\nabla^2 f(x^{k-1})\Delta x^{k-1} - \mathcal{T}_k + E_k \\
&\quad + s^{k-1} + A^T\lambda^{k-1} + \alpha_{k-1}(\Delta s^{k-1} + A^T\Delta\lambda^{k-1}) \\
&= r_c^{k-1} - E_{k-1} - \alpha_{k-1}r_c^{k-1} - \mathcal{T}_k + E_k \\
&= (1-\alpha_{k-1})r_c^{k-1} - \mathcal{T}_k + E_k - E_{k-1}.
\end{aligned}$$

By the definition of the error terms $E_k$ we conclude for all $k \geqslant 0$ that

$$\begin{aligned}
r_c^k &= (1-\alpha_{k-1})r_c^{k-1} - \mathcal{T}_k + E_k - E_{k-1} \\
&= (1-\alpha_{k-1})r_c^{k-1} - \mathcal{T}_k + \sum_{i=0}^{k}\mathcal{T}_i - \sum_{i=0}^{k-1}\mathcal{T}_i \\
&= (1-\alpha_{k-1})r_c^{k-1}.
\end{aligned}$$

Analogously to the proof for $r_b^k$ above we see by induction that

$$r_c^k = (1-\alpha_{k-1})r_c^{k-1} = \prod_{j=0}^{k-1}(1-\alpha_j)r_c^0 = \nu_k r_c^0.$$

Regarding the second part of the theorem, we observe from the the equalities just proven and the fact that all iterates of the algorithm NLIPF are required to belong to the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$ that

$$\nu_k\|(r_b^0, r_c^0)\| = \|(r_b^k, r_c^k)\| \leqslant \frac{\|(r_b^0, r_c^0)\|}{\mu_0}\beta\mu_k.$$

Thus, if $(r_b^0, r_c^0) \neq 0$ then we get that

$$\nu_k \leqslant \frac{\beta}{\mu_0}\mu_k. \qquad \square$$

**Remark 4.2** *In case the residuals $(r_b^0, r_c^0)$ are equal to zero for an initial point $(x^0, \lambda^0, s^0)$ all subsequent iterates $(x^k, \lambda^k, s^k)$ determined by the algorithm NLIPF are strictly feasible. Algorithms exploiting such feasible starting points are discussed for the linear case by Wright [48, chapter 5] and for the convex quadratic case in [13]. However in this thesis we disregard this special case and assume that the initial points have non-zero residuals.*

## 4.4 Assumptions

In the course of the subsequent proof we have to demand certain properties for the objective function and the iterates induced by the algorithm NLIPF. These necessary conditions are listed concisely in this section.

**Assumption 4.1 (Convexity of the objective function)** *Let the scalarised objective functions $f(x) : \mathbb{R}^n \to \mathbb{R}$ be convex, i.e.*

$$(x_1 - x_2)^T(\nabla f(x_1) - \nabla f(x_2)) \geqslant 0$$

*for all $x_1, x_2 \in \mathbb{R}^n$. Compare [20, page 185] for this relation.*

**Assumption 4.2 ("Convexity bound" condition of the objective function)** *The starting point $(x^0, \lambda^0, s^0)$ and the optimal solution $(x^*, \lambda^*, s^*)$ satisfy a "convexity bound" condition with respect to the scalarised objective functions $f(x) : \mathbb{R}^n \to \mathbb{R}$, i.e. there exists a $U \geqslant 0$ such that*

$$(x^0 - x^*)^T(\nabla f(x^0) - \nabla f(x^*)) \leqslant U.$$

**Assumption 4.3 (”Strong” convexity of the objective function)** *Let $(x^0, \lambda^0, s^0)$ be the starting point of our algorithm and $(x^*, \lambda^*, s^*)$ be a solution. All iterates $(x^k, \lambda^k, s^k)$ determined in the course of the algorithm satisfy the following two ”strong” convexity conditions*

$$(x^0 - x^k)^T(\nabla f(x^0) - (\nabla f(x^k) - E_k)) \geqslant 0$$

*and*

$$(x^* - x^k)^T(\nabla f(x^*) - (\nabla f(x^k) - E_k)) \geqslant 0$$

*where $E_k$ are the error terms introduced in the previous section.*

**Assumption 4.4** *The optimal solution $(x^*, \lambda^*, s^*)$ and the starting point of the algorithm NLIPF $(x^0, \lambda^0, s^0)$ are elements of the set*

$$\Phi := \{(x, \lambda, s) \in \mathbb{R}^{n \times m \times n} | (x, s) \geqslant 0, \|(x, s)\|_\infty \leqslant \rho\}$$

*where $\rho \geqslant 0$.*

Hence, if $\|(x, s)\|_\infty \leqslant \rho$ then

$$\begin{aligned} \|x^0 - x^*\|_2 &= (\textstyle\sum_{i=1}^n |x_i^0 - x_i^*|^2)^{1/2} \\ &\leqslant (n\|x^0 - x^*\|_\infty^2)^{1/2} \\ &= \sqrt{n}\|x^0 - x^*\|_\infty \\ &\leqslant \sqrt{n}\rho. \end{aligned} \tag{4.4}$$

**Assumption 4.5 (Lipschitz continuity of the gradient function)** *Let the gradient function $\nabla f(x) : \mathbb{R}^n \to \mathbb{R}^n$ be globally Lipschitz continuous, i.e.*

$$\|\nabla f(x_1) - \nabla f(x_2)\| \leqslant L\|x_1 - x_2\|$$

*for all $x_1, x_2 \in \mathbb{R}^n$ and $L \geqslant 0$.*

**Assumption 4.6** *Let $\mathcal{T}_{k+1} = \mathcal{T}(x^k, \alpha_k \Delta x^k)$ be the Taylor expansion vectors of the gradient functions $\nabla f(x^k + \alpha_k \Delta x^k)$, $\alpha_k$ be the step lengths parameters and $\nu_k$ be the quantities established in the above section. The following inequality is satisfied for all $k \geqslant 0$ and for some $\kappa \geqslant 0$*

$$\|\mathcal{T}_{k+1}\| \leqslant \alpha_k \nu_k \kappa.$$

Prior to the presentation of the proof of the convergence of the algorithm NLIPF we devise key technical results in the next two sections.

## 4.5 Two preliminary results

The deduction of the following two theorems in this section exploits some ideas by Wang & Fei [46] with considerable modifications to adapt to our case.

### 4.5.1 Bounds of the form $\nu_k \|(x^k, s^k)\|_1 \leqslant C_1 \mu_k + C_2$

**Theorem 4.2** *For all iterates $(x^k, \lambda^k, s^k)$ computed by the algorithm NLIPF the following inequalities are satisfied for all $k \geqslant 0$*

$$\nu_k \|(x^k, s^k)\|_1 \leqslant C_1 \mu_k + C_2$$

*where $C_1, C_2$ are nonnegative constants.*

*Proof.* Let $(x^0, \lambda^0, s^0)$ be the starting point of the NLIPF algorithm, $(x^*, \lambda^*, s^*)$ be a solution and $(x^k, \lambda^k, s^k)$ an iterate of the algorithm. We define a new vector by

$$(\overline{x}, \overline{\lambda}, \overline{s}) := \nu_k (x^0, \lambda^0, s^0) + (1 - \nu_k)(x^*, \lambda^*, s^*) - (x^k, \lambda^k, s^k).$$

In addition, we recall the revised definition of the residuals $r_c^k$ and highlight that

$$s^k = \nabla f(x^k) - A^T \lambda^k + r_c^k - E_k.$$

Moreover, we require from a solution $(x^*, \lambda^*, s^*)$ that it satisfies $-\nabla f(x^*) + A^T \lambda^* + s^* = 0$ and thus, obtain that the following equation has to hold

$$r_c^* - E_* = 0.$$

Furthermore, we derive with the definition of the residuals $r_b$ and (4.2) that

$$\begin{aligned} A\overline{x} &= A(\nu_k x^0 + (1 - \nu_k)x^* - x^k) \\ &= \nu_k A x^0 + (1 - \nu_k) A x^* - A x^k \\ &= \nu_k (r_b^0 + b) + (1 - \nu_k) b - (r_b^k + b) \\ &= \nu_k r_b^0 - r_b^k \\ &= 0. \end{aligned}$$

With these observations, (4.2), the facts that $E_0 = \mathcal{T}_0 := 0$ and $0 \leqslant \nu_k \leqslant 1$ and Assumptions 4.1, 4.2 and 4.3 we deduce that

$$
\begin{aligned}
\overline{x}^T\overline{s} &= \overline{x}^T(\nu_k s^0 + (1-\nu_k)s^* - s^k) \\
&= (\nu_k x^0 + (1-\nu_k)x^* - x^k)^T(\nu_k\nabla f(x^0) + (1-\nu_k)\nabla f(x^*) - \nabla f(x^k)) \\
&\quad + \overline{x}^T(-\nu_k A^T\lambda^0 - (1-\nu_k)A^T\lambda^* + A^T\lambda^k) \\
&\quad + \overline{x}^T(\nu_k(r_c^0 - E_0) + (1-\nu_k)(r_c^* - E^*) - (r_c^k - E_k)) \\
&= (\nu_k(x^0 - x^*) + x^* - x^k)^T(\nu_k(\nabla f(x^0) - \nabla f(x^*)) + \nabla f(x^*) - \nabla f(x^k)) \\
&\quad + (\underbrace{A\overline{x}}_{=0})^T(-\nu_k\lambda^0 - (1-\nu_k)\lambda^* + \lambda^k) \\
&\quad + \overline{x}^T(\underbrace{\nu_k r_c^0 - r_c^k}_{=0} - \nu_k \underbrace{E_0}_{=0} + (1-\nu_k)\underbrace{(r_c^* - E^*)}_{=0} + E_k) \\
&= \nu_k^2(x^0 - x^*)^T(\nabla f(x^0) - \nabla f(x^*)) + (x^* - x^k)^T(\nabla f(x^*) - \nabla f(x^k)) \\
&\quad + \nu_k[(x^0 - x^*)^T(\nabla f(x^*) - \nabla f(x^k)) + (x^* - x^k)^T(\nabla f(x^0) - \nabla f(x^*))] \\
&\quad + (\nu_k x^0 + (1-\nu_k)x^* - x^k)^T E_k \\
&= \nu_k^2(x^0 - x^*)^T(\nabla f(x^0) - \nabla f(x^*)) + (x^* - x^k)^T(\nabla f(x^*) - \nabla f(x^k)) \\
&\quad + \nu_k[x^0\nabla f(x^*) - x^0\nabla f(x^k) - x^*\nabla f(x^*) + x^*\nabla f(x^k) + x^*\nabla f(x^0) - x^*\nabla f(x^*) \\
&\qquad - x^k\nabla f(x^0) + x^k\nabla f(x^*) + x^0\nabla f(x^0) - x^0\nabla f(x^0) + x^k\nabla f(x^k) - x^k\nabla f(x^k)] \\
&\qquad + (\nu_k x^0 + (1-\nu_k)x^* - x^k + \nu_k x^k - \nu_k x^k)^T E_k \\
&= -\underbrace{\nu_k(1-\nu_k)}_{\geqslant 0;\ \leqslant 1}\underbrace{(x^0 - x^*)^T(\nabla f(x^0) - \nabla f(x^*))}_{\geqslant 0;\ \leqslant U} \\
&\quad + \nu_k(x^0 - x^k)^T(\nabla f(x^0) - \nabla f(x^k)) + (1-\nu_k)(x^* - x^k)^T(\nabla f(x^*) - \nabla f(x^k)) \\
&\quad + \nu_k(x^0 - x^k)^T E_k + (1-\nu_k)(x^* - x^k)^T E_k \\
&\geqslant -U + \nu_k\underbrace{(x^0 - x^k)^T(\nabla f(x^0) - (\nabla f(x^k) - E_k))}_{\geqslant 0} \\
&\quad + (1-\nu_k)\underbrace{(x^* - x^k)^T(\nabla f(x^*) - (\nabla f(x^k) - E_k))}_{\geqslant 0} \\
&\geqslant -U.
\end{aligned}
$$

Hence, this leads to

$$\begin{aligned}
-U &\leqslant \overline{x}^T\overline{s} \\
&= (\nu_k x^0 + (1-\nu_k)x^* - x^k)^T(\nu_k s^0 + (1-\nu_k)s^* - s^k) \\
&= \nu_k^2 (x^0)^T s^0 + \nu_k(1-\nu_k)[(x^0)^T s^* + (x^*)^T s^0] - \nu_k[(x^0)^T s^k + (x^k)^T s^0] \\
&\quad + (1-\nu_k)^2 \underbrace{(x^*)^T s^*}_{=0} - (1-\nu_k)\underbrace{[(x^*)^T s^k + (x^k)^T s^*]}_{\geqslant 0} + (x^k)^T s^k.
\end{aligned}$$

Since $(x^*, \lambda^*, s^*)$ is a solution we have $(x^*)^T s^* = 0$. In addition, we request $(x^*, s^*) \geqslant 0$ and $(x^k, s^k) > 0$ and therefore $(x^*)^T s^k + (x^k)^T s^* \geqslant 0$ holds. Thus, we can conclude that

$$\nu_k[(x^0)^T s^k + (x^k)^T s^0] \leqslant U + \nu_k^2 (x^0)^T s^0 + \nu_k(1-\nu_k)[(x^0)^T s^* + (x^*)^T s^0] + (x^k)^T s^k. \tag{4.5}$$

Firstly, exploiting the definitions of the norms $\| \cdot \|_1$, $\| \cdot \|_\infty$ and the set $\Phi$ establish in Assumption 4.4 we get

$$\begin{aligned}
(x^0)^T s^* + (x^*)^T s^0 &\leqslant \|x^0\|_\infty \|s^*\|_1 + \|x^*\|_1 \|s^0\|_\infty \\
&\leqslant \max\{\|x^0\|_\infty, \|s^0\|_\infty\} \textstyle\sum_{i=1}^n |x_i^*| + |s_i^*| \\
&\leqslant \|(x^0, s^0)\|_\infty \|(x^*, s^*)\|_1 \\
&\leqslant n\rho^2.
\end{aligned} \tag{4.6}$$

Secondly, let us define a constant $\xi$ by

$$\xi := \min_{i \in \{1,\ldots,n\}} \min\{x_i^0, s_i^0\}.$$

Since $(x^0, s^0) > 0$ it follows that $\xi > 0$. With this definition and the fact that $(x^k, s^k) > 0$

we see that

$$\xi\|(x^k, s^k)\|_1 \leqslant \min_i(s_i^0)\|x^k\|_1 + \min_i(x_i^0)\|s^k\|_1 \leqslant (s^0)^T x^k + (x^0)^T s^k = (x^k)^T s^0 + (x^0)^T s^k. \tag{4.7}$$

Altogether we derive from the inequalities (4.7), (4.5), (4.6), the fact that $0 \leqslant \nu_k \leqslant 1$, the definition of $\mu$ and (4.3) that

$$\begin{aligned}
\nu_k\|(x^k, s^k)\|_1 &\leqslant \xi^{-1}\nu_k[(x^0)^T s^k + (x^k)^T s^0] \\
&\leqslant \xi^{-1}[U + \nu_k^2(x^0)^T s^0 + \nu_k(1-\nu_k)[(x^0)^T s^* + (x^*)^T s^0] + (x^k)^T s^k] \\
&\leqslant \xi^{-1}[U + \nu_k n\mu_0 + \nu_k n\rho^2 + n\mu_k] \\
&\leqslant \xi^{-1}[U + \beta n\mu_k + \frac{\beta}{\mu_0}\mu_k n\rho^2 + n\mu_k] \\
&= C_1\mu_k + C_2
\end{aligned}$$

where $C_1 := \xi^{-1}n[\beta(1 + \frac{\rho^2}{\mu_0}) + 1] \geqslant 0$ and $C_2 := \xi^{-1}U \geqslant 0$. □

### 4.5.2 Bounds on $\|D^{-1}\Delta x^k\|$ and $\|D\Delta s^k\|$

**Theorem 4.3** *In each iteration of the algorithm NLIPF the following inequalities hold*

$$\|D^{-1}\Delta x^k\| \leqslant C_3\mu_k^{1/2} + C_4\mu_k^{-1/2}$$

*and*

$$\|D\Delta s^k\| \leqslant C_3\mu_k^{1/2} + C_4\mu_k^{-1/2}$$

*where $D := X^{1/2}S^{-1/2}$ and $C_3, C_4$ are nonnegative constants.*

*Proof.* Let us consider the following system

$$\begin{bmatrix} A & 0 \\ S & X \end{bmatrix} \begin{bmatrix} \Delta x' \\ \Delta s' \end{bmatrix} = \begin{bmatrix} 0 \\ p \end{bmatrix} \tag{4.8}$$

where $\Delta s' = \nabla f(x + \Delta x') - \nabla f(x) - A^T \Delta\lambda'$ and $p$ is an arbitrary vector of parameters.

Firstly, we observe from the definition of $\Delta s'$, Assumption 4.1 and the first row of system (4.8) that

$$\begin{aligned} (\Delta x')^T \Delta s' &= (\Delta x')^T (\nabla f(x + \Delta x') - \nabla f(x) - A^T \Delta\lambda') \\ &= \underbrace{(x + \Delta x' - x)^T (\nabla f(x + \Delta x') - \nabla f(x))}_{\geqslant 0} - (\underbrace{A\Delta x'}_{=0})^T \Delta\lambda' \\ &\geqslant 0. \end{aligned} \tag{4.9}$$

Furthermore, we derive by multiplying the second row of system (4.8) by $(XS)^{-1/2}$ and recalling the definition of $D := X^{1/2} S^{-1/2}$ that

$$\begin{aligned} S\Delta x' + X\Delta s' &= p \\ D^{-1}\Delta x' + D\Delta s' &= (XS)^{-1/2} p. \end{aligned}$$

Taking squared norms of both sides of the equation and expanding the inner product leads to

$$\begin{aligned} \|D^{-1}\Delta x' + D\Delta s'\|^2 &= \|(XS)^{-1/2} p\|^2 \\ \|D^{-1}\Delta x'\|^2 + 2\,\underbrace{(\Delta x')^T \Delta s'}_{\geqslant 0} + \|D\Delta s'\|^2 &= \|(XS)^{-1/2} p\|^2. \end{aligned}$$

Hence, we can deduce with (4.9) that

$$\begin{aligned} \|D^{-1}\Delta x'\|^2 &\leqslant \|(XS)^{-1/2}p\|^2 \\ \|D^{-1}\Delta x'\| &\leqslant \|(XS)^{-1/2}p\|. \end{aligned} \tag{4.10}$$

And analogously we obtain that

$$\|D\Delta s'\| \leqslant \|(XS)^{-1/2}p\|. \tag{4.11}$$

Let us now examine system (4.8) more closely with the specific vectors

$$\widehat{\Delta x} := \alpha_k(\Delta x^k + \nu_k(x^0 - x^*))$$

and

$$\widehat{\Delta s} := \nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k) - A^T(\alpha_k \Delta\lambda^k).$$

Regarding the first row of system (4.8) we verify that with the definitions of $\widehat{\Delta x}$ and the residuals $r_b^k$, the second row of (4.1) and (4.2) that

$$\begin{aligned} A\widehat{\Delta x} &= \alpha_k(A\Delta x^k + \nu_k A x^0 - \nu_k A x^*) \\ &= \alpha_k(-r_b^k + \nu_k(r_b^0 + b) - \nu_k b) \\ &= \alpha_k(\nu_k r_b^0 - r_b^k) \\ &= 0. \end{aligned}$$

Furthermore, we derive from the definitions of the iterates $(x^k, \lambda^k, s^k)$, the revised residuals

$r_c^k$ and the error terms $E_k$ as well as (4.2) that

$$\begin{aligned}
s^{k+1} &= s^k + \alpha_k \Delta s^k \\
\nabla f(x^{k+1}) - A^T\lambda^{k+1} + r_c^{k+1} - E_{k+1} &= \nabla f(x^k) - A^T\lambda^k + r_c^k - E_k + \alpha_k\Delta s^k \\
\nabla f(x^k + \alpha_k\Delta x^k) - A^T(\lambda^k + \alpha_k\Delta\lambda^k) &= \nabla f(x^k) - A^T\lambda^k + \alpha_k\Delta s^k \\
+(1-\alpha_k)r_c^k - \textstyle\sum_{i=0}^{k+1}\mathcal{T}_i & \quad +r_c^k - \textstyle\sum_{i=0}^{k}\mathcal{T}_i \\
\alpha_k\Delta s^k &= \nabla f(x^k + \alpha_k\Delta x^k) - \nabla f(x^k) - \alpha_k A^T\Delta\lambda^k \\
& \quad -\alpha_k r_c^k - \mathcal{T}_{k+1} \\
&= \nabla f(x^k + \alpha_k\Delta x^k) - \nabla f(x^k) - \alpha_k A^T\Delta\lambda^k \\
& \quad -\alpha_k\nu_k r_c^0 - \mathcal{T}_{k+1}.
\end{aligned} \tag{4.12}$$

Now we can determine $p$ with the definition of $(\widehat{\Delta x}, \widehat{\Delta s})$, the last row of (4.1) and (4.12) as

$$\begin{aligned}
p &= S\widehat{\Delta x} + X\widehat{\Delta s} \\
&= S\alpha_k(\Delta x^k + \nu_k(x^0 - x^*)) + X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k) - \alpha_k A^T \Delta\lambda^k) \\
&= \alpha_k(-XSe + \sigma_k\mu_k e - X\Delta s^k) + \alpha_k\nu_k S(x^0 - x^*) \\
&\quad + X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k) - \alpha_k A^T \Delta\lambda^k) \\
&= \alpha_k(-XSe + \sigma_k\mu_k e) \\
&\quad - X(\nabla f(x^k + \alpha_k\Delta x^k) - \nabla f(x^k) - \alpha_k A^T\Delta\lambda^k - \alpha_k\nu_k r_c^0 - \mathcal{T}_{k+1}) \\
&\quad + \alpha_k\nu_k S(x^0 - x^*) \\
&\quad + X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k) - \alpha_k A^T \Delta\lambda^k) \\
&= \alpha_k(-XSe + \sigma_k\mu_k e) \\
&\quad + \alpha_k\nu_k S(x^0 - x^*) \\
&\quad + X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k)) \\
&\quad + X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1}).
\end{aligned}$$

Next we introduce solutions $(\widehat{\Delta x}_1, \widehat{\Delta s}_1)$, $(\widehat{\Delta x}_2, \widehat{\Delta s}_2)$, $(\widehat{\Delta x}_3, \widehat{\Delta s}_3)$ and $(\widehat{\Delta x}_4, \widehat{\Delta s}_4)$ to system (4.8) with

$$\begin{aligned}
p_1 &= \alpha_k(-XSe + \sigma_k\mu_k e), \\
p_2 &= \alpha_k\nu_k S(x^0 - x^*), \\
p_3 &= X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k)), \\
p_4 &= X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1}).
\end{aligned}$$

We observe an immediate connection between these solutions and $(\widehat{\Delta x}, \widehat{\Delta s})$, namely that

$$\widehat{\Delta x} = \widehat{\Delta x}_1 + \widehat{\Delta x}_2 + \widehat{\Delta x}_3 + \widehat{\Delta x}_4 \tag{4.13}$$

and

$$\widehat{\Delta s} = \widehat{\Delta s}_1 + \widehat{\Delta s}_2 + \widehat{\Delta s}_3 + \widehat{\Delta s}_4. \tag{4.14}$$

Furthermore, we want to highlight that

$$\begin{aligned}
S\widehat{\Delta x}_2 + X\widehat{\Delta s}_2 &= p_2 \\
S\widehat{\Delta x}_2 + X\widehat{\Delta s}_2 &= \alpha_k \nu_k S(x^0 - x^*) \\
D^{-1}\widehat{\Delta x}_2 + D\widehat{\Delta s}_2 &= (XS)^{-1/2}\alpha_k \nu_k S(x^0 - x^*) \\
D\widehat{\Delta s}_2 &= -D^{-1}\widehat{\Delta x}_2 + \alpha_k \nu_k D^{-1}(x^0 - x^*) \\
\|D\widehat{\Delta s}_2\| &= \|D^{-1}(\widehat{\Delta x}_2 - \alpha_k \nu_k (x^0 - x^*)\|.
\end{aligned} \tag{4.15}$$

In addition, we conclude that

$$\begin{aligned}
S\widehat{\Delta x}_3 + X\widehat{\Delta s}_3 &= p_3 \\
S\widehat{\Delta x}_3 + X\widehat{\Delta s}_3 &= X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k \Delta x^k)) \\
D^{-1}\widehat{\Delta x}_3 + D\widehat{\Delta s}_3 &= (XS)^{-1/2} X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k \Delta x^k)) \\
D^{-1}\widehat{\Delta x}_3 &= -D\widehat{\Delta s}_3 + D(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k \Delta x^k))) \\
\|D^{-1}\widehat{\Delta x}_3\| &= \|D(\widehat{\Delta s}_3 - (\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k \Delta x^k)))\|.
\end{aligned} \tag{4.16}$$

With the definition of $\widehat{\Delta x}$, (4.13), the triangular inequality, (4.15), (4.10) and (4.11) we deduce that

$$\begin{aligned}
&\alpha_k\|D^{-1}\Delta x^k\| \\
&= \|D^{-1}\alpha_k\Delta x^k\| \\
&= \|D^{-1}(\widehat{\Delta x} - \alpha_k\nu_k(x^0 - x^*))\| \\
&= \|D^{-1}(\widehat{\Delta x}_1 + \widehat{\Delta x}_2 + \widehat{\Delta x}_3 + \widehat{\Delta x}_4 - \alpha_k\nu_k(x^0 - x^*))\| \\
&\leqslant \|D^{-1}\widehat{\Delta x}_1\| + \|D^{-1}(\widehat{\Delta x}_2 - \alpha_k\nu_k(x^0 - x^*)\| + \|D^{-1}\widehat{\Delta x}_3\| + \|D^{-1}\widehat{\Delta x}_4\| \qquad (4.17)\\
&= \|D^{-1}\widehat{\Delta x}_1\| + \|D\widehat{\Delta s}_2\| + \|D^{-1}\widehat{\Delta x}_3\| + \|D^{-1}\widehat{\Delta x}_4\| \\
&\leqslant \|(XS)^{-1/2}\alpha_k(-XSe + \sigma_k\mu_k e)\| + \|(XS)^{-1/2}\alpha_k\nu_k S(x^0 - x^*)\| \\
&\quad + \|(XS)^{-1/2}X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k))\| \\
&\quad + \|(XS)^{-1/2}X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1})\|.
\end{aligned}$$

Moreover, we see with the definition of $\widehat{\Delta x}$, Assumptions 4.5 and 4.6 and (4.4) that

$$\begin{aligned}
&\alpha_k\|D^{-1}\Delta x^k\| \\
&\leqslant \alpha_k\|(XS)^{-1/2}\|\|-XSe + \sigma_k\mu_k e\| + \alpha_k\nu_k\|(XS)^{-1/2}\|\|S\|\|x^0 - x^*\| \\
&\quad + \|(XS)^{-1/2}\|\|X\|\|\nabla f(x^k + \alpha_k\Delta x^k + \alpha_k\nu_k(x^0 - x^*)) - \nabla f(x^k + \alpha_k\Delta x^k)\| \\
&\quad + \|(XS)^{-1/2}\|\|X\|\|\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1}\| \\
&\leqslant \alpha_k\|(XS)^{-1/2}\|\|-XSe + \sigma_k\mu_k e\| + \alpha_k\nu_k\|(XS)^{-1/2}\|\|S\|\|x^0 - x^*\| \\
&\quad + \|(XS)^{-1/2}\|\|X\|L\alpha_k\nu_k\|x^0 - x^*\| + \|(XS)^{-1/2}\|\|X\|(\alpha_k\nu_k\|r_c^0\| + \|\mathcal{T}_{k+1}\|) \\
&\leqslant \alpha_k\|(XS)^{-1/2}\|\Big[\|-XSe + \sigma_k\mu_k e\| + \nu_k\|S\|\sqrt{n}\rho + \nu_k\|X\|L\sqrt{n}\rho + \nu_k\|X\|(\|r_c^0\| + \kappa)\Big].
\end{aligned} \qquad (4.18)$$

Here we want to point out that with the requirements of the neighbourhood $\mathcal{N}_{-\infty}(\gamma,\beta)$ it follows that

$$\|(XS)^{-1/2}\| = \max_{i=1,\dots,n}\frac{1}{(x_i^k s_i^k)^{1/2}} \leqslant \frac{1}{(\gamma\mu_k)^{1/2}}. \qquad (4.19)$$

Secondly, we deduce with the relations between norms ($\|x\|_2 \leqslant \|x\|_1$ for all $x \in \mathbb{R}^n$), the definition of $\mu_k$ and the fact that $\sigma_k \in (0,1)$ that

$$\begin{aligned}
\| -XSe + \sigma_k\mu_k e\|^2 &= \|XSe\|^2 - 2\sigma_k\mu_k (x^k)^T s^k + \sigma_k^2\mu_k^2 n \\
&\leqslant \|XSe\|_1^2 - 2\sigma_k n\mu_k^2 + \sigma_k n\mu_k^2 \\
&= ((x^k)^T s^k)^2 - \underbrace{\sigma_k n\mu_k^2}_{\geqslant 0} \\
&\leqslant n^2\mu_k^2.
\end{aligned} \tag{4.20}$$

Therefore we obtain that

$$\| -XSe + \sigma_k\mu_k e\| \leqslant n\mu_k. \tag{4.21}$$

Finally, we conclude from the relations between norms ($\|x\|_\infty \leqslant \|x\|_1$ for all $x \in \mathbb{R}^n$) and Theorem 4.2 that

$$\nu_k\|S\| = \nu_k(\max_{i=1,\dots,n} |s_i^k|^2)^{1/2} = \nu_k\|s^k\|_\infty \leqslant \nu_k\|s^k\|_1 \leqslant \nu_k\|(x^k,s^k)\|_1 \leqslant C_1\mu_k + C_2 \tag{4.22}$$

respectively

$$\nu_k\|X\| = \nu_k(\max_{i=1,\dots,n} |x_i^k|^2)^{1/2} = \nu_k\|x^k\|_\infty \leqslant \nu_k\|x^k\|_1 \leqslant \nu_k\|(x^k,s^k)\|_1 \leqslant C_1\mu_k + C_2. \tag{4.23}$$

Thus, we derive by dividing inequality (4.18) by $\alpha_k$ and (4.19), (4.21), (4.22), (4.23) that

$$\begin{aligned}
&\|D^{-1}\Delta x^k\| \\
&\leqslant \|(XS)^{-1/2}\| \Big[\| -XSe + \sigma_k\mu_k e\| + \nu_k\|S\|\sqrt{n}\rho + \nu_k\|X\|L\sqrt{n}\rho + \nu_k\|X\|(\|r_c^0\| + \kappa)\Big] \\
&\leqslant \frac{1}{(\gamma\mu_k)^{1/2}}\Big[n\mu_k + (C_1\mu_k + C_2)\sqrt{n}\rho + (C_1\mu_k + C_2)L\sqrt{n}\rho + (C_1\mu_k + C_2)(\|r_c^0\| + \kappa)\Big] \\
&= \widetilde{C}_3\mu_k^{1/2} + \widetilde{C}_4\mu_k^{-1/2}
\end{aligned}$$

where $\widetilde{C}_3 := \gamma^{-1/2}[n + C_1(\sqrt{n}\rho(1+L) + (\|r_c^0\| + \kappa))]$ and $\widetilde{C}_4 := \gamma^{-1/2}C_2(\sqrt{n}\rho(1+L) +$

$(\|r_c^0\| + \kappa))$.

In addition, we scrutinise the result (4.12) further and obtain with the definition of $\widehat{\Delta s}$ that

$$\begin{aligned}
&\alpha_k \Delta s^k \\
&= \nabla f(x^k + \alpha_k \Delta x^k) - \nabla f(x^k) - \alpha_k A^T \Delta\lambda^k - \alpha_k \nu_k r_c^0 - \mathcal{T}_{k+1} \\
&= \nabla f(x^k + \alpha_k \Delta x^k) + \widehat{\Delta s} - \nabla f(x^k + \widehat{\Delta x}) - \alpha_k \nu_k r_c^0 - \mathcal{T}_{k+1} \\
&= \widehat{\Delta s} - (\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k \Delta x^k)) - \alpha_k \nu_k r_c^0 - \mathcal{T}_{k+1}.
\end{aligned} \tag{4.24}$$

Hence, it follows from (4.24), (4.14), the triangular inequality, (4.16), the definition of $D$, (4.10) and (4.11) that

$$\begin{aligned}
&\alpha_k\|D\Delta s^k\| \\
&= \|D\alpha_k\Delta s^k\| \\
&= \|D(\widehat{\Delta s} - (\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k)) - \alpha_k\nu_k r_c^0 - \mathcal{T}_{k+1})\| \\
&= \|D(\widehat{\Delta s}_1 + \widehat{\Delta s}_2 + \widehat{\Delta s}_3 + \widehat{\Delta s}_4 - (\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k)) \\
&\quad - \alpha_k\nu_k r_c^0 - \mathcal{T}_{k+1})\| \\
&\leqslant \|D\widehat{\Delta s}_1\| + \|D\widehat{\Delta s}_2\| + \|D(\widehat{\Delta s}_3 - (\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k)))\| \\
&\quad + \|D\widehat{\Delta s}_4\| + \|D(-\alpha_k\nu_k r_c^0 - \mathcal{T}_{k+1})\| \\
&= \|D\widehat{\Delta s}_1\| + \|D\widehat{\Delta s}_2\| + \|D^{-1}\widehat{\Delta x}_3\| + \|D\widehat{\Delta s}_4\| + \|(XS)^{-1/2}X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1})\| \\
&\leqslant \|(XS)^{-1/2}\alpha_k(-XSe + \sigma_k\mu_k e)\| + \|(XS)^{-1/2}\alpha_k\nu_k S(x^0 - x^*)\| \\
&\quad + \|(XS)^{-1/2}X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k))\| \\
&\quad + \|(XS)^{-1/2}X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1})\| + \|(XS)^{-1/2}X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1})\| \\
&= \|(XS)^{-1/2}\alpha_k(-XSe + \sigma_k\mu_k e)\| + \|(XS)^{-1/2}\alpha_k\nu_k S(x^0 - x^*)\| \\
&\quad + \|(XS)^{-1/2}X(\nabla f(x^k + \widehat{\Delta x}) - \nabla f(x^k + \alpha_k\Delta x^k))\| \\
&\quad + 2\|(XS)^{-1/2}X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1})\|.
\end{aligned}$$

We point out that this upper bound differs from the previous case (4.17) only by the additional term $\|(XS)^{-1/2}X(\alpha_k\nu_k r_c^0 + \mathcal{T}_{k+1})\|$. Hence, we derive in similar fashion that

$$\|D\Delta s^k\| \leqslant C_3\mu_k^{1/2} + C_4\mu_k^{-1/2}$$

where $C_3 := \gamma^{-1/2}[n + C_1(\sqrt{n}\rho(1+L) + 2(\|r_c^0\| + \kappa))]$ and $C_4 := \gamma^{-1/2}C_2(\sqrt{n}\rho(1+L) + 2(\|r_c^0\| + \kappa))$.

Furthermore, we observe that

$$\begin{aligned}
\widetilde{C}_3 &= \gamma^{-1/2}[n + C_1(\sqrt{n}\rho(1+L) + (\|r_c^0\| + \kappa))] \\
&\leqslant \gamma^{-1/2}[n + C_1(\sqrt{n}\rho(1+L) + (\|r_c^0\| + \kappa))] + \gamma^{-1/2}C_1(\|r_c^0\| + \kappa) \\
&= C_3
\end{aligned}$$

and

$$\begin{aligned}
\widetilde{C}_4 &= \gamma^{-1/2}C_2(\sqrt{n}\rho(1+L) + (\|r_c^0\| + \kappa)) \\
&\leqslant \gamma^{-1/2}C_2(\sqrt{n}\rho(1+L) + (\|r_c^0\| + \kappa)) + \gamma^{-1/2}C_2(\|r_c^0\| + \kappa) \\
&= C_4.
\end{aligned}$$

Thus, the inequality

$$\|D^{-1}\Delta x^k\| \leqslant C_3\mu_k^{1/2} + C_4\mu_k^{-1/2}$$

also holds, which completes the proof. □

## 4.6 Two technical results

With these preliminary results at our disposal we can proceed by deriving bounds for the scalar product of the solution vectors of system (4.1).

### 4.6.1 Bounds on $|(\Delta x^k)^T \Delta s^k|$

**Theorem 4.4** *For the step length vectors $\Delta x^k$ and $\Delta s^k$ determined by the algorithm NLIPF, the following inequalities hold for all $k \geqslant 0$*

$$|(\Delta x^k)^T \Delta s^k| \leqslant (C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2.$$

*And particularly,*

$$-(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 \leqslant (\Delta x^k)^T \Delta s^k.$$

*Proof.* Utilising the Cauchy-Schwarz inequality and Theorem 4.3 we observe that

$$\begin{aligned} |(\Delta x^k)^T \Delta s^k| &= |(D^{-1}\Delta x^k)^T (D\Delta s^k)| \\ &\leqslant \|D^{-1}\Delta x^k\| \|D\Delta s^k\| \\ &\leqslant (C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2. \end{aligned}$$ □

**Theorem 4.5** *For single entries of the step length vectors $\Delta x^k$ and $\Delta s^k$ determined by the algorithm NLIPF, the following inequalities hold for all $k \geqslant 0$ and all $i = 1, \ldots, n$*

$$|\Delta x_i^k \Delta s_i^k| \leqslant (C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2.$$

*And in particular,*

$$-(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 \leqslant \Delta x_i^k \Delta s_i^k.$$

*Proof.* Again, exploiting the Cauchy-Schwarz inequality and Theorem 4.3 it follows that

$$\begin{aligned} |\Delta x_i^k \Delta s_i^k| &= |D_{ii}^{-1}\Delta x_i^k||D_{ii}\Delta s_i^k| \\ &\leqslant \|D^{-1}\Delta x^k\| \|D\Delta s^k\| \\ &\leqslant (C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2. \end{aligned}$$ □

### 4.6.2 Bound on $(\Delta x^k)^T \Delta s^k$

In the following section we establish an improved upper bound the scalar product $(\Delta x^k)^T \Delta s^k$.

**Theorem 4.6** *For the step length vectors $\Delta x^k$ and $\Delta s^k$ computed by the algorithm NLIPF, the following inequalities hold for all $k \geqslant 0$*

$$(\Delta x^k)^T \Delta s^k \leqslant \frac{n^2}{2\gamma}\mu_k.$$

*Proof.* From the last row of (4.1) we obtain that

$$S\Delta x^k + X\Delta s^k = -XSe + \sigma_k \mu_k e.$$

Multiplying this equation by $(XS)^{-1/2}$ and recalling that $D := S^{-1/2}X^{1/2}$ leads to

$$D^{-1}\Delta x^k + D\Delta s^k = (XS)^{-1/2}(-XSe + \sigma_k \mu_k e).$$

Taking squared norms of both sides of the equation and expanding the inner product results in

$$\begin{array}{rcl} \|D^{-1}\Delta x^k + D\Delta s^k\|^2 & = & \|(XS)^{-1/2}(-XSe + \sigma_k\mu_k e)\|^2 \\ \underbrace{\|D^{-1}\Delta x^k\|^2}_{\geqslant 0} + 2(\Delta x^k)^T\Delta s^k + \underbrace{\|D\Delta s^k\|^2}_{\geqslant 0} & \leqslant & (\|(XS)^{-1/2}\| \| - XSe + \sigma_k\mu_k e\|)^2 \\ (\Delta x^k)^T \Delta s^k & \leqslant & \frac{1}{2}\|(XS)^{-1/2}\|^2 \| - XSe + \sigma_k\mu_k e\|^2. \end{array}$$

With (4.19) and (4.20) we conclude that

$$(\Delta x^k)^T\Delta s^k \leqslant \frac{1}{2}\|(XS)^{-1/2}\|^2\| - XSe + \sigma_k\mu_k e\|^2 \leqslant \frac{1}{2}(\frac{1}{(\gamma\mu_k)^{1/2}})^2 n^2 \mu_k^2 = \frac{n^2}{2\gamma}\mu_k. \qquad \square$$

## 4.7 Lower bounds on $\alpha_k$

The next theorem establishes a lower bound for all step lengths parameters $\alpha_k$.

**Theorem 4.7** *Let $k \geqslant 0$ and $(x^k, \lambda^k, s^k)$ be vectors induced by the algorithm NLIPF. Then there exists a value $\overline{\alpha_k} \in (0,1)$ such that the following conditions are met for all $\alpha \in [0, \overline{\alpha_k}]$:*

- $(x^k + \alpha \Delta x^k)^T (s^k + \alpha \Delta s^k) \geqslant (1-\alpha)(x^k)^T s^k$,
- $(x_i^k + \alpha \Delta x_i^k)^T (s_i^k + \alpha \Delta s_i^k) \geqslant \frac{\gamma}{n}(x^k + \alpha \Delta x^k)^T (s^k + \alpha \Delta s^k)$,
- $(x^k + \alpha \Delta x^k)^T (s^k + \alpha \Delta s^k) \leqslant (1 - 0.01\alpha)(x^k)^T s^k$.

*Proof.* Before we start with the proof of the three statements, we highlight two expedient observations.

Firstly, one single component of the last row of system (4.1) reads

$$s_i^k \Delta x_i^k + x_i^k \Delta s_i^k = -x_i^k s_i^k + \sigma_k \mu_k. \tag{4.25}$$

Secondly, summing over all $n$ components of this equation results in

$$\begin{aligned}(s^k)^T \Delta x^k + (x^k)^T \Delta s^k &= -(x^k)^T s^k + n\sigma_k \mu_k \\ &= -(x^k)^T s^k + \sigma_k (x^k)^T s^k \\ &= (\sigma_k - 1)(x^k)^T s^k.\end{aligned} \tag{4.26}$$

Concerning the first statement, we deduce with (4.26), the fact that $\sigma_k \geqslant \sigma_{min}$ for all $k \geqslant 0$ and Theorem 4.4 that

$$\begin{aligned}(x^k + \alpha_k \Delta x^k)^T (s^k + \alpha_k \Delta s^k) &= (x^k)^T s^k + \alpha_k ((\Delta x^k)^T s^k + (x^k)^T \Delta s^k) + \alpha_k^2 (\Delta x^k)^T \Delta s^k \\ &= (x^k)^T s^k + \alpha_k (\sigma_k - 1)(x^k)^T s^k + \alpha_k^2 (\Delta x^k)^T \Delta s^k \\ &\geqslant (1-\alpha_k)(x^k)^T s^k + \alpha_k \sigma_{min} (x^k)^T s^k - \alpha_k^2 (C_3 \mu_k^{1/2} + C_4 \mu_k^{-1/2})^2.\end{aligned}$$

Hence the first condition is satisfied if

$$0 \leqslant \alpha_k \sigma_{min} (x^k)^T s^k - \alpha_k^2 (C_3 \mu_k^{1/2} + C_4 \mu_k^{-1/2})^2,$$

i.e.,

$$\begin{aligned}\alpha_k &\leqslant \frac{\sigma_{min}(x^k)^T s^k}{(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2} \\ &= \frac{\sigma_{min} n \mu_k}{(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2}.\end{aligned}$$

Regarding the second inequality, we derive with (4.25), Theorem 4.5 and the definition of the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$ for the left-hand side that

$$\begin{aligned}(x_i^k + \alpha_k \Delta x_i^k)^T (s_i^k + \alpha_k \Delta s_i^k) &= x_i^k s_i^k + \alpha_k (\Delta x_i^k s_i^k + x_i^k \Delta s_i^k) + \alpha_k^2 \Delta x_i^k \Delta s_i^k \\ &= x_i^k s_i^k + \alpha_k (-x_i^k s_i^k + \sigma_k \mu_k) + \alpha_k^2 \Delta x_i^k \Delta s_i^k \\ &\geqslant (1 - \alpha_k) x_i^k s_i^k + \alpha_k \sigma_k \mu_k - \alpha_k^2 (C_3 \mu_k^{1/2} + C_4 \mu_k^{-1/2})^2 \\ &\geqslant (1 - \alpha_k) \gamma \mu_k + \alpha_k \sigma_k \mu_k - \alpha_k^2 (C_3 \mu_k^{1/2} + C_4 \mu_k^{-1/2})^2.\end{aligned}$$

For the right-hand side we obtain with (4.26), Theorem 4.6 and the definition of $\mu_k$ that

$$\begin{aligned}\frac{\gamma}{n}(x^k + \alpha_k \Delta x^k)^T (s^k + \alpha_k \Delta s^k) &= \frac{\gamma}{n}((x^k)^T s^k + \alpha_k((\Delta x^k)^T s^k + (x^k)^T \Delta s^k) + \alpha_k^2 (\Delta x^k)^T \Delta s^k) \\ &= \frac{\gamma}{n}((x^k)^T s^k + \alpha_k(\sigma_k - 1)(x^k)^T s^k + \alpha_k^2 (\Delta x^k)^T \Delta s^k) \\ &\leqslant \frac{\gamma}{n}((x^k)^T s^k + \alpha_k(\sigma_k - 1)(x^k)^T s^k + \alpha_k^2 \frac{n^2}{2\gamma} \mu_k) \\ &= \frac{\gamma}{n}((1 - \alpha_k)(x^k)^T s^k + \alpha_k \sigma_k (x^k)^T s^k + \frac{\alpha_k^2 n^2}{2\gamma} \mu_k) \\ &= (1 - \alpha_k) \gamma \mu_k + \alpha_k \gamma \sigma_k \mu_k + \frac{\alpha_k^2 n}{2} \mu_k.\end{aligned}$$

Taking the differences between both sides and recalling that $\sigma_k \geqslant \sigma_{min}$ for all $k \geqslant 0$ leads to

$$
\begin{aligned}
&(x_i^k + \alpha_k \Delta x_i^k)^T (s_i^k + \alpha_k \Delta s_i^k) - \frac{\gamma}{n}(x^k + \alpha_k \Delta x^k)^T (s^k + \alpha_k \Delta s^k) \\
&\geqslant (1-\alpha_k)\gamma\mu_k + \alpha_k\sigma_k\mu_k - \alpha_k^2 (C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 - (1-\alpha_k)\gamma\mu_k - \alpha_k\gamma\sigma_k\mu_k - \frac{\alpha_k^2 n}{2}\mu_k \\
&= (1-\gamma)\alpha_k\sigma_k\mu_k - \alpha_k^2[(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 + \frac{n}{2}\mu_k] \\
&\geqslant (1-\gamma)\alpha_k\sigma_{min}\mu_k - \alpha_k^2[(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 + \frac{n}{2}\mu_k].
\end{aligned}
$$

Hence, the second statement holds if

$$0 \leqslant (1-\gamma)\alpha_k\sigma_{min}\mu_k - \alpha_k^2[(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 + \frac{n}{2}\mu_k]$$

i.e.,

$$\alpha_k \leqslant \frac{(1-\gamma)\sigma_{min}\mu_k}{(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 + \frac{n}{2}\mu_k}.$$

Finally, we observe with (4.26), Theorem 4.6, the definition of $\mu_k$ and the fact that we assume $\sigma_k \leqslant \sigma_{max} < 0.5$ for the differences between both sides of the third condition divided by $n$ that

$$
\begin{aligned}
&\frac{1}{n}((x^k + \alpha_k \Delta s^k)^T (s^k + \alpha_k \Delta s^k) - (1 - 0.01\alpha_k)(x^k)^T s^k) \\
&= \frac{1}{n}((x^k)^T s^k + \alpha_k((\Delta x^k)^T s^k + (x^k)^T \Delta s^k) + \alpha_k^2 (\Delta x^k)^T \Delta s^k) - (1 - 0.01\alpha_k)(x^k)^T s^k) \\
&\leqslant \frac{1}{n}((1-\alpha_k)(x^k)^T s^k + \alpha_k\sigma_k (x^k)^T s^k + \frac{\alpha_k^2 n^2}{2\gamma}\mu_k - (1 - 0.01\alpha_k)(x^k)^T s^k) \\
&= (1-\alpha_k)\mu_k + \alpha_k\sigma_k\mu_k + \frac{\alpha_k^2 n}{2\gamma}\mu_k - (1-0.01\alpha_k)\mu_k \\
&= -0.99\alpha_k\mu_k + \alpha_k\sigma_k\mu_k + \frac{\alpha_k^2 n}{2\gamma}\mu_k \\
&\leqslant -0.99\alpha_k\mu_k + \alpha_k 0.5\mu_k + \frac{\alpha_k^2 n}{2\gamma}\mu_k \\
&= (-0.49\alpha_k + \frac{\alpha_k^2 n}{2\gamma})\mu_k.
\end{aligned}
$$

Hence the third inequality holds if

$$n(-0.49\alpha_k + \frac{\alpha_k^2 n}{2\gamma})\underbrace{\mu_k}_{\geqslant 0} \leqslant 0$$

i.e.,

$$\begin{aligned} \frac{\alpha_k^2 n}{2\gamma} &\leqslant 0.49\alpha_k \\ \alpha_k &\leqslant \frac{0.98\gamma}{n}. \end{aligned}$$

Combining these three bounds, we conclude that the three conditions are satisfied if $\alpha \in [0, \overline{\alpha_k}]$ where

$$\overline{\alpha_k} := \min[\frac{\sigma_{min} n \mu_k}{(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2}, \frac{(1-\gamma)\sigma_{min}\mu_k}{(C_3\mu_k^{1/2} + C_4\mu_k^{-1/2})^2 + \frac{n}{2}\mu_k}, \frac{0.98\gamma}{n}]. \tag{4.27}$$

□

**Remark 4.3** *The algorithm NLIPF computes a maximal step length $\alpha_k \in [0,1]$. As a consequence of Theorem 4.7 these parameters are at least as great as $\overline{\alpha_k}$ in each iteration, i.e. $\alpha_k \geqslant \overline{\alpha_k} > 0$. The essential conclusion from these positive step lengths is, that the algorithm NLIPF advances substantially at each single iteration.*

We proceed by deriving from the above theorem that the algorithm NLIPF is well defined.

**Theorem 4.8** *Let $(x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k))$ be an iterate generated by the algorithm NLIPF. Then*

$$(x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k)) \in \mathcal{N}_{-\infty}(\gamma, \beta)$$

*and the sufficient decrease inequality*

$$0 \leqslant \mu_k(\alpha_k) \leqslant (1 - 0.01\alpha_k)\mu_k$$

*holds.*

*Proof.* We employ the estimates from Theorem 4.7. Starting from the iterate $(x^k, \lambda^k, s^k)$ it follows from the first condition in Theorem 4.7 that

$$\mu_k(\alpha_k) = \frac{(x^k + \alpha_k \Delta x^k)^T (s^k + \alpha_k \Delta s^k)}{n} \geqslant \frac{(1-\alpha_k)(x^k)^T s^k}{n} = (1-\alpha_k)\mu_k. \tag{4.28}$$

Consequently, we use Theorem 4.1, (4.28) and the fact that $(x^k, \lambda^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$ to derive that

$$\|(r_b^k(\alpha_k), r_c^k(\alpha_k))\| = (1-\alpha_k)\|(r_b^k, r_c^k)\| \leqslant \frac{\mu_k(\alpha_k)}{\mu_k}\|(r_b^k, r_c^k)\| \leqslant \frac{\|(r_b^0, r_c^0)\|}{\mu_0}\beta\mu_k(\alpha_k).$$

From the second condition we directly obtain that

$$x_i^k(\alpha_k)s_i^k(\alpha_k) \geqslant \gamma\mu_k(\alpha_k).$$

Hence, we have proven that the new iterate $(x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k))$ is an element of the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$.

Due to the third condition we deduce that

$$0 \leqslant \mu_k(\alpha_k) = \frac{(x^k + \alpha_k \Delta x^k)^T (s^k + \alpha_k \Delta s^k)}{n} \leqslant \frac{(1-0.01\alpha_k)(x^k)^T s^k}{n} \leqslant (1-0.01\alpha_k)\mu_k$$

and hence, the sufficient decrease condition is satisfied as well. □

We recapitulate that we have proven that our algorithm NLIPF is well-defined and that it achieves a reduction in $\mu$ in every single iteration, i.e. $\mu_{k+1} \leqslant \mu_k$. Furthermore, by the construction of the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$ we notice that also the residuals $r_b$ and $r_c$ are decreased during every step of the algorithm, since they are bounded by a multiple of

$\mu_k$.

The last step that remains in the proof of convergence is to show that $\mu_k \to 0$, which we derive in the final theorem of this chapter.

**Theorem 4.9** *The "duality gaps" of the iterates $(x^k, \lambda^k, s^k)$ of our algorithm NLIPF converge to zero, i.e.*

$$\mu_k \to 0 \quad \textit{for } k \to \infty.$$

*Proof.* We prove this theorem via contradiction. Hence, let us assume that $\mu_k$ does not converge to 0. Then there exists an $\varepsilon_1 > 0$ and some $k^* \geqslant 0$ such that

$$\mu_k \geqslant \varepsilon_1 > 0 \qquad \text{for all } k \geqslant k^*.$$

Consequently we derive from (4.27) that there exists an $\varepsilon_2 > 0$ such that

$$\overline{\alpha_k} \geqslant \varepsilon_1 > 0 \quad \text{for all } k \geqslant k^*.$$

With the explanations provided in Remark 4.3 we deduce that

$$\alpha_k \geqslant \varepsilon_2 > 0 \text{ for all } k \geqslant k^*$$

and thus, the following expression converges to zero for $k \to \infty$

$$\prod_{i=0}^{k} (1 - 0.01\alpha_i) \to 0.$$

Utilising the sufficient decrease condition from Theorem 4.8 we observe that

$$0 \leqslant \mu_k \leqslant \underbrace{\prod_{i=0}^{k} (1 - 0.01\alpha_i)}_{\to 0} \mu_0 = 0$$

which contradicts our assumption of the divergence of the series $\mu_k$. Hence, we have proven that the series $\mu_k$ does converge to 0. $\square$

# Chapter 5

# Warm-Start Strategy

In this chapter we introduce the warm-start strategy which plays a crucial part in the process of approximating the whole Pareto-optimal set for a multiobjective optimisation problem. Initially we outline the underlying concept of this technique which is followed by a preliminary examination involving Taylor expansion analysis. Subsequently we constitute the criteria that a newly computed iterate has to meet and establish an essential condition that is utilised to detect the admissibility of potential candidates. We complete our review of this subject area by scrutinising the adjustments made to the neighbourhood of the central path for the newly generated iterate.

The framework of this elegant warm-start approach was introduced by Yildirim and Wright [49] for the linear programming case and Heermann [18] developed an enhancement for the case of convex quadratic objective functions. In addition, the paper by John and Yildirim [23] details several warm-start strategies for linear programming. We apply their ideas to the general framework of multiobjective optimisation problems featuring nonlinear functions. In the following sections we describe the inherent difficulties this extension entails and describe our new contributions and analysis.

## 5.1 Idea of the warm-start strategy

So far the previous chapters focussed on determining a solution for one particular instance of the scalarised ersatz problems with a uniquely defined weighting parameter $\omega$. However, in order to obtain a meaningful approximation of the Pareto-optimal set we have to develop a method that identifies many solutions to several different ersatz problems with varying weighting parameters. Hence, given an iterate $(x, \lambda, s) \in \mathcal{N}_{-\infty}(\gamma, \beta)$ of the algorithm NLIPF for the problem instance characterised by $\omega$ the idea of the warm-start strategy is to compute a new positive iterate $(\tilde{x}, \tilde{\lambda}, \tilde{s})$, which we denote as warm-start iterate, for a related problem instance with a different weighting parameter $\tilde{\omega}$. Furthermore, we require that this new iterate also lies in a modified neighbourhood of the central path $\widetilde{\mathcal{N}}_{-\infty}(\tilde{\gamma}, \tilde{\beta})$ with new parameters which are specified in the course of our investigation.

Yildirim and Wright as well as Heermann can identify the objective functions in their discussions via a single vector $c$ in the linear case or one matrix $Q$ and one vector $c$ in the convex quadratic case. Consequently, they can exploit data triplets $(A, b, c)$ and data quadruples $(A, b, Q, c)$ to describe the optimisation problems in their analysis and derive conditions for feasible perturbations based on this structure. In our general nonlinear setting we cannot follow this path and have to construct an alternative way to formalise the warm-start method. Hence, we develop our new warm-start strategy by exploiting Taylor expansion analysis for different weighting parameters which we present in the next section. Subsequently, we illustrate the mechanism of the warm-start strategy in more details.

## 5.2 Taylor analysis

Let $\omega$ and $\tilde{\omega} := \omega + \Delta\omega$ be the weighting parameters used to formulate the objective functions of the scalarised ersatz problems. We recapitulate that in the bi-criteria case these objective functions, the corresponding Jacobian vectors and Hessian matrices are of the form

$$\begin{aligned} f(x) &= \omega f_1(x) + (1-\omega) f_2(x) \\ \nabla f(x) &= \omega \nabla f_1(x) + (1-\omega)\nabla f_2(x) && (5.1) \\ \nabla^2 f(x) &= \omega \nabla^2 f_1(x) + (1-\omega)\nabla^2 f_2(x) && (5.2) \end{aligned}$$

and similarly for weighting parameter $\tilde{\omega}$

$$\begin{aligned} \widetilde{f}(x) &= \tilde{\omega} f_1(x) + (1-\tilde{\omega}) f_2(x) \\ \nabla \widetilde{f}(x) &= \tilde{\omega} \nabla f_1(x) + (1-\tilde{\omega})\nabla f_2(x) \\ \nabla^2 \widetilde{f}(x) &= \tilde{\omega} \nabla^2 f_1(x) + (1-\tilde{\omega})\nabla^2 f_2(x). \end{aligned}$$

Furthermore, we recall the definition of the Taylor expansion introduced in Appendix B where $\mathcal{T}(x, \Delta x)$ is the short hand notation for the vector of the remaining Taylor expansion terms

$$f(x + \Delta x) = f(x) + \nabla f(x)\Delta x + \mathcal{T}(x, \Delta x).$$

Applying this concept to the Jacobian vector

$$\nabla f(x + \Delta x) = \omega \nabla f_1(x + \Delta x) + (1-\omega)\nabla f_2(x + \Delta x)$$

we deduce that

$$\begin{aligned}
\nabla f(x) + \nabla^2 f(x)\Delta x + \mathcal{T}(x,\Delta x) &= \omega\nabla f_1(x) + \omega\nabla^2 f_1(x)\Delta x + \omega\mathcal{T}_1(x,\Delta x) \\
&\quad +(1-\omega)\nabla f_2(x) + (1-\omega)\nabla^2 f_2(x)\Delta x \\
&\quad +(1-\omega)\mathcal{T}_2(x,\Delta x)
\end{aligned}$$

where $\mathcal{T}(x,\Delta x), \mathcal{T}_1(x,\Delta x)$ and $\mathcal{T}_2(x,\Delta x)$ correspond to the Taylor expansion vectors of the functions $\nabla f(x+\Delta x), \nabla f_1(x+\Delta x)$ and $\nabla f_2(x+\Delta x)$, respectively. Utilising (5.1) and (5.2) we derive from the above equation that

$$\mathcal{T}(x,\Delta x) = \omega\mathcal{T}_1(x,\Delta x) + (1-\omega)\mathcal{T}_2(x,\Delta x).$$

With this result at hand we observe for the vector of the remaining Taylor expansion terms of the function $\nabla\widetilde{f}(x+\Delta x)$, which we refer to as $\widetilde{\mathcal{T}}(x,\Delta x)$, that

$$\begin{aligned}
\widetilde{\mathcal{T}}(x,\Delta x) &= \tilde{\omega}\mathcal{T}_1(x,\Delta x) + (1-\tilde{\omega})\mathcal{T}_2(x,\Delta x) \\
&= (\omega+\Delta\omega)\mathcal{T}_1(x,\Delta x) + (1-(\omega+\Delta\omega))\mathcal{T}_2(x,\Delta x) \\
&= \omega\mathcal{T}_1(x,\Delta x) + (1-\omega)\mathcal{T}_2(x,\Delta x) \qquad (5.3) \\
&\quad +\Delta\omega(\mathcal{T}_1(x,\Delta x) - \mathcal{T}_2(x,\Delta x)) \\
&= \mathcal{T}(x,\Delta x) + \Delta\omega(\mathcal{T}_1(x,\Delta x) - \mathcal{T}_2(x,\Delta x)).
\end{aligned}$$

With these technical results at our disposal we are able to proceed and concentrate on the necessary conditions that a new warm-start iterate has to satisfy in the subsequent section.

## 5.3 Necessary conditions of new iterates

The criteria that the warm-start iterate has to meet in order to be an admissible point comprise of three areas. In particular we have to examine the residuals and the 'duality gap' (which we define below) of the new iterate as well as analyse the neighbourhood of the central path to which the warm-start iterate has to belong to. For that purpose let $(x, \lambda, s)$ be an iterate for the problem instance constituted by $\omega$ and an element of the neighbourhood $\mathcal{N}_{-\infty}(\gamma, \beta)$. Our aim is to determine a new positive iterate which is denoted by

$$(\tilde{x}, \tilde{\lambda}, \tilde{s}) := (x, \lambda, s) + (\Delta x, \Delta\lambda, \Delta s)$$

for a different problem instance with weighting parameter $\tilde{\omega} := \omega + \Delta\omega$. Before we introduce our modified neighbourhood let us recall the definition of the variable $\mu := (x^T s)/n$ from the previous chapter which defines the 'duality gap' in this examination. Consequently, for a warm-start iterate it is defined as $\tilde{\mu} := (\tilde{x}^T \tilde{s})/n$ and similarly, as $\mu_0 := (x_0^T s_0)/n$ for our unique starting point. Now we label the residuals for this warm-start iterate as $(\widetilde{r_b}, \widetilde{r_c})$ and can state the modified neighbourhood with parameters $\tilde{\beta} \geqslant 1$ and $\tilde{\gamma} \in (0, 1)$ as

$$\begin{aligned}\widetilde{\mathcal{N}}_{-\infty}(\tilde{\gamma}, \tilde{\beta}) \quad := \quad & \{(\tilde{x}, \tilde{\lambda}, \tilde{s}) \mid \|(\widetilde{r_b}, \widetilde{r_c})\| \leqslant \frac{\|(\widetilde{r_b}^0, \widetilde{r_c}^0)\|}{\mu_0}\tilde{\beta}\tilde{\mu}, (\tilde{x}, \tilde{s}) > 0, \\ & \tilde{x}_i\tilde{s}_i \geqslant \tilde{\gamma}\tilde{\mu} \text{ for all } i = 1, \dots, n\}.\end{aligned}$$

In the following discussion we identify the properties that the perturbation vector $(\Delta x, \Delta\lambda, \Delta s)$ has to satisfy in order to produce admissible warm-start iterates.

### 5.3.1 Residuals

The first requirement that has to be fulfilled is that the infeasibility of the warm-start iterate remains the same. Hence, we demand that

$$(\widetilde{r_b}, \widetilde{r_c}) := (r_b, r_c)$$

which results in the conclusion that the following two conditions that have to hold.

$$\begin{aligned} A\tilde{x} - b &= r_b \\ -\nabla \widetilde{f}(\tilde{x}) + A^T\tilde{\lambda} + \tilde{s} + E_k &= r_c. \end{aligned}$$

Regarding the first one we see that

$$r_b = A\tilde{x} - b = A(x + \Delta x) - b = Ax - b + A\Delta x = r_b + A\Delta x.$$

Thus, the subsequent equation has to be satisfied for the computation of the new iterate

$$A\Delta x = 0. \tag{5.4}$$

Concerning the second criterium we conclude with the definitions of the residuals, the Jacobian vectors and the Hessian matrices, the Taylor expansion analysis and (5.3) that

$$
\begin{aligned}
r_c &= -\nabla\widetilde{f}(\tilde{x}) + A^T\tilde{\lambda} + \tilde{s} + E_k \\
&= -\nabla\widetilde{f}(x+\Delta x) + A^T(\lambda+\Delta\lambda) + (s+\Delta s) + E_k \\
&= -\tilde{\omega}\nabla f_1(x+\Delta x) - (1-\tilde{\omega})\nabla f_2(x+\Delta x) + A^T\lambda + s + E_k + A^T\Delta\lambda + \Delta s \\
&= -\omega\nabla f_1(x+\Delta x) - \Delta\omega\nabla f_1(x+\Delta x) \\
&\quad -(1-\omega)\nabla f_2(x+\Delta x) + \Delta\omega\nabla f_2(x+\Delta x) \\
&\quad +A^T\lambda + s + E_k + A^T\Delta\lambda + \Delta s \\
&= -\nabla f(x+\Delta x) - \Delta\omega\nabla f_1(x+\Delta x) + \Delta\omega\nabla f_2(x+\Delta x) \\
&\quad +A^T\lambda + s + E_k + A^T\Delta\lambda + \Delta s \\
&= -\nabla f(x) - \nabla^2 f(x)\Delta x - \mathcal{T}(x,\Delta x) + A^T\lambda + s + E_k + A^T\Delta\lambda + \Delta s \\
&\quad -\Delta\omega(\nabla f_1(x) + \nabla^2 f_1(x)\Delta x + \mathcal{T}_1(x,\Delta x)) \\
&\quad +\Delta\omega(\nabla f_2(x) + \nabla^2 f_2(x)\Delta x + \mathcal{T}_2(x,\Delta x)) \\
&= r_c - \omega\nabla^2 f_1(x)\Delta x - (1-\omega)\nabla^2 f_2(x)\Delta x - \mathcal{T}(x,\Delta x) + A^T\Delta\lambda + \Delta s \\
&\quad -\Delta\omega(\nabla f_1(x) + \nabla^2 f_1(x)\Delta x + \mathcal{T}_1(x,\Delta x)) \\
&\quad +\Delta\omega(\nabla f_2(x) + \nabla^2 f_2(x)\Delta x + \mathcal{T}_2(x,\Delta x)) \\
&= r_c - \tilde{\omega}\nabla^2 f_1(x)\Delta x - (1-\tilde{\omega})\nabla^2 f_2(x)\Delta x - \mathcal{T}(x,\Delta x) + A^T\Delta\lambda + \Delta s \\
&\quad -\Delta\omega(\nabla f_1(x) - \nabla f_2(x)) - \Delta\omega(\mathcal{T}_1(x,\Delta x) - \mathcal{T}_2(x,\Delta x)) \\
&= r_c - \nabla^2\widetilde{f}(x)\Delta x + A^T\Delta\lambda + \Delta s \; - \Delta\omega(\nabla f_1(x) - \nabla f_2(x)) - \widetilde{\mathcal{T}}(x,\Delta x).
\end{aligned}
$$

From this result we deduce that the following equation has to hold to guarantee that the infeasibility of the warm-start iterate has not worsened

$$
-\nabla^2\widetilde{f}(x)\Delta x + A^T\Delta\lambda + \Delta s - \Delta\omega(\nabla f_1(x) - \nabla f_2(x)) - \widetilde{\mathcal{T}}(x,\Delta x) = 0. \tag{5.5}
$$

### 5.3.2 'Duality gap'

In addition to the infeasibility of the warm-start iterate we also desire that the 'duality gap' should not be increased. That is, we insist that the subsequent condition is satisfied

$$\tilde{\mu} \leqslant \mu.$$

A closer look at the definition of these terms reveals the following two equations

$$\mu = \frac{x^T s}{n}$$

and

$$\begin{aligned} \tilde{\mu} &= \frac{\tilde{x}^T \tilde{s}}{n} \\ &= \frac{(x + \Delta x)^T (s + \Delta s)}{n} \\ &= \frac{x^T s + X\Delta s + S\Delta x + \Delta x^T \Delta s}{n}. \end{aligned}$$

Consequently, the required inequality holds if

$$X\Delta s + S\Delta x + \Delta x^T \Delta s \leqslant 0.$$

In the next theorem we scrutinise an equation and show that it exactly implies the above condition.

**Theorem 5.1** *Let $x, s, \Delta x, \Delta s \in \mathbb{R}^n$, $X = diag(x), S = diag(s) \in \mathbb{R}^{n\times n}$ and $(x, s) > 0$. If*

$$X\Delta s + S\Delta x = 0$$

*is satisfied then the three subsequent inequalities hold*

- $\Delta x_i \Delta s_i \leqslant 0$ *for all* $i = 1, \ldots, n$
- $\Delta x^T \Delta s \leqslant 0$
- $X\Delta s + S\Delta x + \Delta x^T \Delta s \leqslant 0.$

*Proof.* If we consider every single line of the equation we obtain that

$$x_i \Delta s_i + s_i \Delta x_i = 0 \quad \text{for all } i = 1, \ldots, n.$$

Since we assume $(x, s) > 0$ it follows that either $\Delta s_i \leqslant 0$ or $\Delta x_i \leqslant 0$ and hence,

$$\Delta x_i \Delta s_i \leqslant 0 \quad \text{for all } i = 1, \ldots, n$$

which proves the first result.

Summing over all indices leads to the second result

$$\Delta x^T \Delta s = \sum_{i=1}^{n} \underbrace{\Delta x_i \Delta s_i}_{\leqslant 0} \leqslant 0.$$

Finally, we observe that

$$\underbrace{X\Delta s + S\Delta x}_{=0} + \underbrace{\Delta x^T \Delta s}_{\leqslant 0} \leqslant 0$$

which completes the theorem. □

Regarding the 'duality gap' condition we derive with the third inequality from the above theorem that the equation

$$X\Delta s + S\Delta x = 0 \tag{5.6}$$

implies that the requirement $\tilde{\mu} \leqslant \mu$ is fulfilled.

### 5.3.3 Warm-start system of equations

From the above investigation we conclude that in addition to the strict non-negativity conditions $(\tilde{x}, \tilde{s}) > 0$ the equations (5.4), (5.5) and (5.6) have to be satisfied in order to compute the perturbation vector $(\Delta x, \Delta\lambda, \Delta s)$ which determines the new warm-start iterate. These requirements can be rewritten in structured form as

$$\begin{bmatrix} -\nabla^2 \widetilde{f}(x) & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} \Delta\omega(\nabla f_1(x) - \nabla f_2(x)) + \widetilde{\mathcal{T}}(x, \Delta x) \\ 0 \\ 0 \end{bmatrix} \tag{5.7}$$

This system of linear equations resembles close similarity to the Newton-step system and in a similar fashion we deduce that the explicit solutions are

$$\begin{aligned} \Delta\lambda &= (AD^2\tilde{B}A^T)^{-1}AD^2\tilde{B}(\Delta\omega(\nabla f_1(x) - \nabla f_2(x)) + \widetilde{\mathcal{T}}(x, \Delta x)) \\ \Delta s &= \tilde{B}(\Delta\omega(\nabla f_1(x) - \nabla f_2(x)) + \widetilde{\mathcal{T}}(x, \Delta x) - A^T\Delta\lambda) \\ \Delta x &= -D^2\Delta s \end{aligned}$$

where $D := S^{-1/2}X^{1/2}$ and $\tilde{B} := (I_n + \nabla^2\widetilde{f}D^2)^{-1}$.

However, contrary to the Newton-step system we cannot introduce a step length parameter for this particular system of equations. Thus, due to the fact that full step lengths do not necessarily create iterates that lie within the neighbourhood $\widetilde{\mathcal{N}}_{-\infty}(\tilde{\gamma}, \tilde{\beta})$ we have to establish a criterium that guarantees the admissibility of the computed iterates.

In the subsequent theorem we provide such a condition.

**Theorem 5.2** *Let $(x, \lambda, s) \in \mathcal{N}_{-\infty}(\gamma, \beta)$ be an iterate of the algorithm NLIPF and let $(\Delta x, \Delta\lambda, \Delta s)$ be a solution to system (5.7). Then the warm-start iterate $(\tilde{x}, \tilde{\lambda}, \tilde{s}) =$*

*$(x, \lambda, s) + (\Delta x, \Delta\lambda, \Delta s)$ is admissible if*

$$\|X^{-1}\Delta x\|_\infty = \|S^{-1}\Delta s\|_\infty < 1.$$

*Proof.* From the earlier explanations in this section we get that the four subsequent requirements have to be satisfied

- $A\tilde{x} - b = r_b$
- $-\nabla\widetilde{f}(\tilde{x}) + A^T\tilde{\lambda} + \tilde{s} = r_c$
- $\tilde{\mu} \leqslant \mu$
- $(\tilde{x}, \tilde{s}) > 0$

for an iterate $(\tilde{x}, \tilde{\lambda}, \tilde{s})$ to qualify as admissible. We obtain immediately from the assumption that $(\Delta x, \Delta\lambda, \Delta s)$ solves system (5.7) and the previous analysis above that the first three conditions hold. Therefore we only need to verify that $(\tilde{x}, \tilde{s}) > 0$. In full details this restriction can be written as

$$\tilde{x} = x + \Delta x > 0 \quad \text{and} \quad \tilde{s} = s + \Delta s > 0$$

and since we assumed $(x, s) > 0$ this is equivalent to the strict inequalities

$$\frac{\Delta x_i}{x_i} > -1 \quad \text{and} \quad \frac{\Delta s_i}{s_i} > -1 \quad \text{for all } i = 1, \ldots, n. \tag{5.8}$$

From the last row of system (5.7) we see that

$$s_i\Delta x_i + x_i\Delta s_i = 0 \quad \text{for all } i = 1, \ldots, n$$

which can be transformed to

$$\frac{\Delta x_i}{x_i} + \frac{\Delta s_i}{s_i} = 0 \quad \text{for all } i = 1, \ldots, n. \tag{5.9}$$

Combining the results (5.8) and (5.9) we obtain that

$$-\frac{\Delta s_i}{s_i} = \frac{\Delta x_i}{x_i} > -1 \quad \text{and} \quad -\frac{\Delta x_i}{x_i} = \frac{\Delta s_i}{s_i} > -1$$

has to hold for all $i = 1, \ldots, n$, which subsequently implies

$$\frac{\Delta s_i}{s_i} < 1 \quad \text{and} \quad \frac{\Delta x_i}{x_i} < 1 \tag{5.10}$$

has to be satisfied for all $i = 1, \ldots, n$. Consequently, it follows from (5.8) and (5.10) that the following inequalities have to hold for all indices $i = 1, \ldots, n$

$$\left|\frac{\Delta x_i}{x_i}\right| < 1 \quad \text{and} \quad \left|\frac{\Delta s_i}{s_i}\right| < 1.$$

Using concise vector norm notation and result (5.9) we derive that

$$\|X^{-1}\Delta x\|_\infty = \|S^{-1}\Delta s\|_\infty < 1$$

is a necessary and sufficient condition for the strict feasibility of the warm-start iterate $(\tilde{x}, \tilde{\lambda}, \tilde{s})$. □

With this result at hand we are now capable of verifying the admissibility of an iterate $(\tilde{x}, \tilde{\lambda}, \tilde{s})$ solely by the criteria $\|S^{-1}\Delta s\|_\infty < 1$. Nevertheless, we still need to examine if these iterates lie within a modified neighbourhood $\widetilde{\mathcal{N}}_{-\infty}(\tilde{\gamma}, \tilde{\beta})$.

## 5.4 Neighbourhood $\widetilde{\mathcal{N}}_{-\infty}(\tilde{\gamma}, \tilde{\beta})$

In order to show that the newly determined warm-start iterate is a member of the set $\widetilde{\mathcal{N}}_{-\infty}(\tilde{\gamma}, \tilde{\beta})$ we introduce the perturbation parameter $\theta \in (0, 1)$ and specify the parameters $\tilde{\beta}, \tilde{\gamma}$ in the next theorem.

**Theorem 5.3** *Let $(x, \lambda, s)$ and $(\Delta x, \Delta\lambda, \Delta s)$ be defined as in Theorem 5.2 and furthermore, let $\theta \in (0, 1)$. If*

$$\|X^{-1}\Delta x\|_\infty = \|S^{-1}\Delta s\|_\infty \leqslant 1 - \theta \tag{5.11}$$

*then*

$$(\tilde{x}, \tilde{\lambda}, \tilde{s}) \in \widetilde{\mathcal{N}}_{-\infty}(\theta\gamma, \theta^{-1}\beta).$$

*Proof.* We observe from the introduction of the parameter $\theta$ that for all indices $i = 1, \ldots, n$

$$\begin{aligned} \frac{\Delta x_i}{x_i} &\geqslant -(1-\theta) \\ \Delta x_i &\geqslant -x_i + \theta x_i \\ \Delta x_i + x_i &\geqslant \theta x_i \end{aligned}$$

is fulfilled and thus, we obtain that

$$\tilde{x}_i = x_i + \Delta x_i \geqslant \theta x_i$$

and similarly

$$\tilde{s}_i = s_i + \Delta s_i \geqslant \theta s_i.$$

In addition, we know by Theorem 5.1 that $\Delta x_i \Delta s_i \leqslant 0$ for all $i = 1, \ldots, n$.
Assume now that $\Delta x_i \geqslant 0$ then $\tilde{x}_i \geqslant x_i$ and subsequently we have

$$\tilde{x}_i\tilde{s}_i \geqslant x_i\tilde{s}_i \geqslant \theta x_i s_i.$$

Analogously, we deduce the same result for the case $\Delta s_i \geqslant 0$ and therefore, conclude that

$$\tilde{x}_i \tilde{s}_i \geqslant \theta x_i s_i \geqslant \theta\gamma\mu \geqslant \theta\gamma\tilde{\mu}$$

for all $i = 1, \ldots, n$.

Secondly, we can show that

$$\tilde{\mu} = \sum_{i=1}^{n} \tilde{x}_i \tilde{s}_i \geqslant \theta \sum_{i=1}^{n} x_i s_i \geqslant \theta\mu$$

and consequently the following relation holds

$$\|(\widetilde{r_b}, \widetilde{r_c})\| = \|(r_b, r_c)\| \leqslant \frac{\|(r_b^0, r_c^0)\|}{\mu_0}\beta\mu \leqslant \frac{\|(r_b^0, r_c^0)\|}{\mu_0}\beta\theta^{-1}\tilde{\mu}.$$

Hence, we have proven that if the warm-start iterate satisfies the assumption then all the conditions that are required to be an element of the set $\widetilde{\mathcal{N}}_{-\infty}(\theta\gamma, \theta^{-1}\beta)$ are met. □

We summarise that we presented the warm-start strategy that exploits existing iterates of the algorithm NLIPF in order to determine new iterates for different weighting parameters. In particular we compute a solution vector to system (5.7) for a chosen $\Delta\omega$. If this solution satisfies the condition (5.11) it epitomises an admissible iterate for a different weighting parameter $\tilde{\omega} = \omega + \Delta\omega$ with respect to the scalarised ersatz problem. In addition, this new warm-start iterate is a member of the modified central path neighbourhood $\widetilde{\mathcal{N}}_{-\infty}(\theta\gamma, \theta^{-1}\beta)$.

The advantage of computing iterates utilising this warm-start strategy rather than initiating the algorithm NLIPF for every weighting parameter individually is that the computational effort is reduced and a broad approximation of the Pareto-optimal set can be achieved early on in the course of the algorithm. The reason for the reduced numerical complexity is that current iterates of the algorithm NLIPF possess better properties with respect to infeasibility and the 'duality gap' than the initial starting point. A more detailed analysis about the operations of the entire solution method as well as the determination of $\Delta\omega$ are provided in the next chapter on our numerical implementation.

# Chapter 6

# Numerical Implementation

The focus of this chapter is on merging the techniques originating from multiobjective optimisation with the interior-point algorithm NLIPF and the warm-start strategy which perturbs current iterates in order to generate new ones for different weighting parameters. To begin with we revise the most important aspects of our method which is followed by the formal formulation of the NL_EFFTREE algorithm. Subsequently, we provide a thorough description of our numerical implementation utilising the software programme MATLAB. In particular we present the data format we employ in the computer code and discuss several features of our implementation.

## 6.1 Brief recap of the algorithm

The aim of our solution method is to determine an accurate and comprehensive approximation of the efficient set for multiobjective optimisation problems of the form

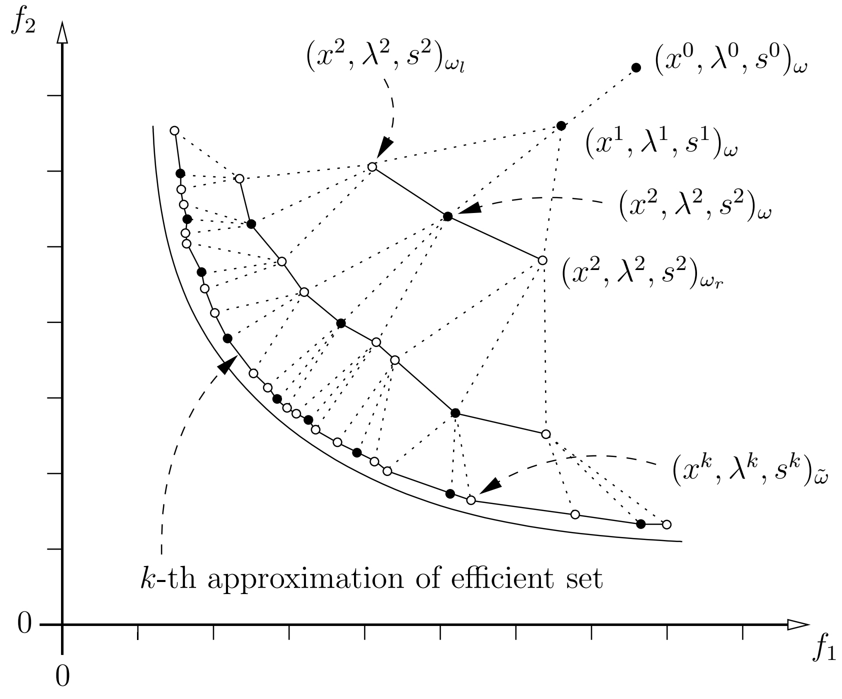
$$\begin{array}{lccl} \text{"min"} & \begin{pmatrix} f_1(x) \\ f_2(x) \end{pmatrix} & & \\ \text{s.t.} & Ax & = & b \\ & x & \geqslant & 0, \end{array} \tag{6.1}$$

where $f_1(x)$,$f_2(x)$ are non-linear twice-differentiable functions $\mathbb{R}^n \to \mathbb{R}$ and $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^n$. With regard to the multiobjective aspect of this problem we followed a widely-used route and provided a scalarisation method in Chapter 2 to compute the efficient set for (6.1) via scalar ersatz problems of the form

$$\begin{array}{lccc} \min & \omega f_1(x) + (1-\omega) f_2(x) & & \\ \text{s.t.} & Ax & = & b \\ & x & \geqslant & 0, \end{array} \tag{6.2}$$

where $\omega \in [0,1]$. We note that problem instances of type (6.2) are uniquely identified by their corresponding weighting parameter $\omega$ and therefore, we also refer to instances $\omega$. In the susequent chapters we devised an infeasible-path-following interior-point algorithm to solve these ersatz problems for particular instances. Afterwards in Chapter 5 we formulated a warm-start strategy to exploit information from iterates for a certain instance that are already computed in order to generate iterates for different problem instances via a perturbation mechanism.

At this point we want to highlight a distinct feature of our algorithm, namely the employment of infeasible iterates. The advantage of this technique is that it is considerably easier to select a suitable initial point from which we can start the process. In fact, it is sufficient to opt for $(x^0, \lambda^0, s^0) = \zeta(e, 0, e)$ where $\zeta > 0$ is chosen sufficient big since the side constraints do not have to be strictly satisfied. Typically the search for an admissible starting point poses a difficult problem for feasible interior-point methods which is precisely the reason why we select our infeasibility approach. Based on that starting point we proceed with the warm-start strategy immediately. That is, we attempt to generate a comprehensive approximation of the efficient set at an early stage of the computation with iterates that do not represent optimal solutions with regard to their related instance yet. This approach promises greater perturbations which imply a broad approximation of the efficient set long before the duality gaps of the individual problem instances are minimised to a desired level. We scrutinise this concept in greater detail below and begin by presenting a graphical illustration of this method in Figure 6.1.



$f_2$
$(x^2, \lambda^2, s^2)_{\omega_l}$
$(x^0, \lambda^0, s^0)_{\omega}$
$(x^1, \lambda^1, s^1)_{\omega}$
$(x^2, \lambda^2, s^2)_{\omega}$
$(x^2, \lambda^2, s^2)_{\omega_r}$
$(x^k, \lambda^k, s^k)_{\tilde{\omega}}$
$k$-th approximation of efficient set
0
0
$f_1$

Figure 6.1: Graphical illustration of the operation of the warm-start strategy.

Initially, the starting point linked to weighting parameter $\omega_0 \in (0, 1)$ forms the first approximation of the efficient set. Now, we perform a fixed number $i_0$ of iterations of the algorithm NLIPF and based on the last iterate we compute perturbation weights $\Delta\omega_l < 0$ and $\Delta\omega_r > 0$, which determine the warm-start iterates for the instances $\tilde{\omega}_l := \omega + \Delta\omega_l$ and $\tilde{\omega}_r := \omega + \Delta\omega_r$. Afterwards, we perform $i_k$ iterations of the algorithm NLIPF for all points. Assuming the warm-start searches were successful, this process lead to an approximation of the efficient set consisting of three points. If the Pareto-optimal curve is convex then the corresponding weighting parameters can be ordered according to size and we label two consecutive parameters as neighbouring instances. The warm-start search and the subsequent $i_k$ iterations of the algorithm NLIPF are repeated for all iterates until the duality gap of every iterate is below a specified level and a predetermined density of the function values in the image space is reached. Since the improving progress of the approximations resembles a tree-like structure in the image space we denote our algorithm NL_EFFTREE. The exact determination of the perturbation weights $\Delta\omega$ will be discussed in later, but beforehand we introduce the formal notation of the algorithm in the following section.

## 6.2 Algorithm NL_EFFTREE

Before we can establish a formal description of the algorithm NL_EFFTREE we want to recapitulate the parameters that are utilised by the algorithm NLIPF for sake of completeness. In particular, these are the parameters $\beta$ and $\gamma$ of the neigbourhood of the central path as well as the centering parameter $\sigma$ and the parameter $\varepsilon$ which specifies the tolerance with regard to the maximum duality gap that we want to permit. In addition, we constituted the parameter $\theta$ which is used during the warm-start search to amend the neighbourhoods for the new warm-start iterates.

Furthermore, we need to introduce an additional parameter, denoted by $\delta$, to quantify the distance between the function values of two iterates in the image space. Let $(x^k, \lambda^k, s^k)_\omega$ be the $k$-th iterate for the instance $\omega$ and define the function

$$\begin{array}{rccc} \varphi_k : & [0,1] & \rightarrow & f(\Omega) \subset \mathbb{R}^2 \\ & \omega & \rightarrow & f(x^k_\omega) \end{array}$$

where $x^k_\omega$ represents the primal variable of the iterate $(x^k, \lambda^k, s^k)_\omega$. We exploit the Euclidean norm to determine the distance between the image points of two iterates $\omega$, $\widehat{\omega}$ as

$$\|\varphi_k(\omega) - \varphi_k(\widehat{\omega})\|_2.$$

Moreover, to ease the handling of all the different instances of the scalarised ersatz problems we introduce the set $\mathcal{W}$ which contains all weighting parameters for which the algorithm has already computed iterates. In addition, we introduce the notion of the auxiliary set $\widehat{\mathcal{W}}$, a subset of $\mathcal{W}$, which comprises of all weighting parameter instances that have been checked for a warm-start during each iteration of the algorithm as well as the new warm-start iterates that have been computed during this iteration. In more details, this set $\widehat{\mathcal{W}}$ is empty at the outset of every iteration and at the end contains all weighting parameters of the previously computed iterates plus the ones of the newly computed warm-start iterates. Finally, we define the neighbouring instances for an iterate $\omega \in \mathcal{W}$ as

$$\begin{aligned} \omega_l &:= \max\big\{\omega_l \in \mathcal{W} \cup \{0,1\} | \omega_l < \omega\big\}, \\ \omega_r &:= \min\big\{\omega_r \in \mathcal{W} \cup \{0,1\} | \omega_r > \omega\big\} \end{aligned}$$

and save them in the set $\mathcal{W}_\omega := \{\omega_l, \omega_r\} \subset \mathcal{W}$. The purpose of this definition is that from a graphical perspective, under the assumption of a convex Pareto front, these instances represent the iterate paths exactly to the right and left of the current iterate.

The number of iterations of the algorithm NLIPF that are carried out during the $k^{th}$ loop of the algorithm NL_EFFTREE is defined as $i_k$ $(k \geqslant 0)$. For simplicity, we set the values $i_0 := l$ and $i_k := l'$ for all $k \geqslant 1$ where $l, l' \in \mathbb{N}$.

We conclude this section by summarising all variables and parameters that are used during the course of the algorithm (except the warm-start parameters which we examine later on in this chapter) in Table 6.1 supplemented by a brief description, which is followed by a concise formulation of the algorithm NL_EFFTREE.

**Remark 6.1** *We want to highlight that the loop counter $k$ of the algorithm NL_EFFTREE does not reflect the actual number of iterations of NLIPF computed for an iterate $(x^k, \lambda^k, s^k)$.*

| Parameter | Description |
|---|---|
| $k$ | loop counter |
| $\varepsilon$ | maximal duality gap |
| $\delta$ | maximal distance between images points |
| $\zeta$ | starting point parameter |
| $e$ | unity vector $(1, \dots, 1)^T \in \mathbb{R}^n$ |
| $\omega$ | weighting parameter |
| $\Delta\omega$ | perturbation strength |
| $\mathcal{W}$ | set of all computed instances |
| $\widehat{\mathcal{W}}$ | set of all instances checked for warm-starts |
| $\mathcal{W}_\omega$ | set of all neighbouring instances for $\omega$ |
| $i_k$ | iterations of NLIPF in loop $k$ |
| $\alpha$ | step length multiplier in NLIPF |
| $\sigma_\omega$ | centering parameter |
| $\beta_\omega, \gamma_\omega$ | neighbourhood parameters |

Table 6.1: List of parameters

*Rather, this depth is determined by the sum of iterations* $\sum_{j=0}^{k} i_j$ *computed so far.*

**Algorithm NL_EFFTREE**

```
• Initialisation phase

  choose $\omega_{initial} \in [0,1]$, $\zeta > 0$, $i_k \in \mathbb{N}$ $(k = 0, 1, 2, \dots)$, $\beta_{initial} \geqslant 1$, $\gamma_{initial} \in (0,1)$,
      $\sigma_{initial} \in (0, 0.5)$, $\varepsilon > 0$, $\delta > 0$, $\theta \in (0,1)$

  set $k := 0$, $(x^0, \lambda^0, s^0)_{\omega_{initial}} := \zeta(e, 0, e)$, $\mathcal{W} := \{\omega_{initial}\}$, $\beta_\omega := \beta_{initial}$, $\gamma_\omega := \gamma_{initial}$,
      $\sigma_\omega := \sigma_{initial}$

• Iteration phase

  for all $\omega \in \mathcal{W}$

  compute $(x^{k+1}, \lambda^{k+1}, s^{k+1})_\omega$ by performing $i_k$ iterations of the algorithm NLIPF
  starting from $(x^k, \lambda^k, s^k)_\omega$ and employing parameters $\beta_\omega, \gamma_\omega, \sigma_\omega$

  end (for all)

• Warm-start search phase

  set $\widehat{\mathcal{W}} := \emptyset$

  while $\mathcal{W} \backslash \widehat{\mathcal{W}} \neq \emptyset$

    – choose $\omega \in \mathcal{W} \backslash \widehat{\mathcal{W}}$

    – define $\omega_l := \max\{\omega_l \in \mathcal{W} \cup \{0,1\} | \omega_l < \omega\}$,
             $\omega_r := \min\{\omega_r \in \mathcal{W} \cup \{0,1\} | \omega_r > \omega\}$

    – for all $\widehat{\omega} \in \mathcal{W}_\omega := \{\omega_l, \omega_r\}$:
      if $\|\varphi_k(\omega_l) - \varphi_k(\omega)\|_2 > \delta$ or $\|\varphi_k(\omega_r) - \varphi_k(\omega)\|_2 > \delta$
```

```
            * determine $\Delta\omega \in \big(\omega_l - \omega, 0\big]$ or $\big[0, \omega_r - \omega\big)$ with $|\Delta\omega|$ maximal, such that
              $\Delta\omega$ is a solution of the system

              $$\begin{bmatrix} -\nabla^2 \widetilde{f}(x^{k+1}) & A^T & I \\ A & 0 & 0 \\ S_\omega^{k+1} & 0 & X_\omega^{k+1} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} \Delta\omega(\nabla f_1(x^{k+1}) - \nabla f_2(x^{k+1})) \\ 0 \\ 0 \end{bmatrix} \tag{6.3}$$

              and the point generated by

              $$(x^{k+1}, \lambda^{k+1}, s^{k+1})_{\tilde{\omega}} := (x^{k+1}, \lambda^{k+1}, s^{k+1})_\omega + (\Delta x, \Delta\lambda, \Delta s)$$

              is an element of the neighbourhood set $\mathcal{N}_{-\infty}(\gamma_{\tilde{\omega}}, \beta_{\tilde{\omega}})$, where $\tilde{\omega} := \omega + \Delta\omega$,
              $\gamma_{\tilde{\omega}} := \theta\gamma_\omega$ and $\beta_{\tilde{\omega}} := \beta_\omega/\theta$
            * if $\Delta\omega = 0$, then set

              $$\tilde{\omega} := \omega + 0.4(\widehat{\omega} - \omega)$$

              and determine $(x^{k+1}, \lambda^{k+1}, s^{k+1})_{\tilde{\omega}}$ as a result of a cold start using start-
              ing point $(x^0, \lambda^0, s^0)_{\tilde{\omega}} := \zeta(e, 0, e)$, parameters $\beta_{initial}$, $\gamma_{initial}$, $\sigma_{initial}$ and
              performing $\sum_{j=0}^{k+1} i_j$ iterations of the algorithm NLIPF
            * set $\mathcal{W} := \mathcal{W} \cup \{\tilde{\omega}\}, \widehat{\mathcal{W}} := \widehat{\mathcal{W}} \cup \{\tilde{\omega}\}$
        – end (for all)
        – set $\widehat{\mathcal{W}} := \widehat{\mathcal{W}} \cup \{\omega\}$
    **end (while)**
• Abort criterion
    **if** $\big(\max\{\mu_\omega\} < \varepsilon$ **and** $\max\{\|\varphi_k(\omega) - \varphi_k(\widehat{\omega})\|_2\} < \delta,\ \forall \widehat{\omega} \in \mathcal{W}_\omega\big)\ \forall \omega \in \mathcal{W}$ STOP
    otherwise set $k := k + 1$
        GOTO 'Iteration phase'
```

**Remark 6.2** *At this point we want to remark that we neglect the Taylor expansion terms of the original warm-start system (5.7) in the linear system (6.3) of our numerical algorithm for sake of simplicity.*

## 6.3 Implementation in MATLAB

We decided that MATLAB represents the appropriate computer software to serve as a platform for the implementation of our algorithm NL_EFFTREE. To provide the reader with a better understanding of the operations of our code and the interactions between the different routines we depict the relations of all employed M-files in the following diagrams Figure 6.2 and Figure 6.3. Here, we want to point out that the routines `create_warm_start`, `create_cold_start` and `NLIPF_algorithm` are listed in both pictures and are in fact referring to identical M-files. Figure 6.2 is illustrating the head of the algorithm as well as the interface to the mathematical core of the code (via the three M-files mentioned above). In addition, Figure 6.3 shows how the routines that are responsible for the actual determination of the iterates and the computation of the first and second-order derivatives are associated.

**Remark 6.3** *We note that the files* `TP_Ab` *and* `TP_eff_set` *in Figure 6.2 as well as the file* `TP_f_values` *in Figure 6.3 are M-files which contain specific information characterising the test problem we want to compute. More details about this particular structure are provided in the next chapter about test problems.*

In order to separate the crucial M-files from auxiliary routines we present the significant routines in the subsequent Table 6.2 and provide a short explanation about their key

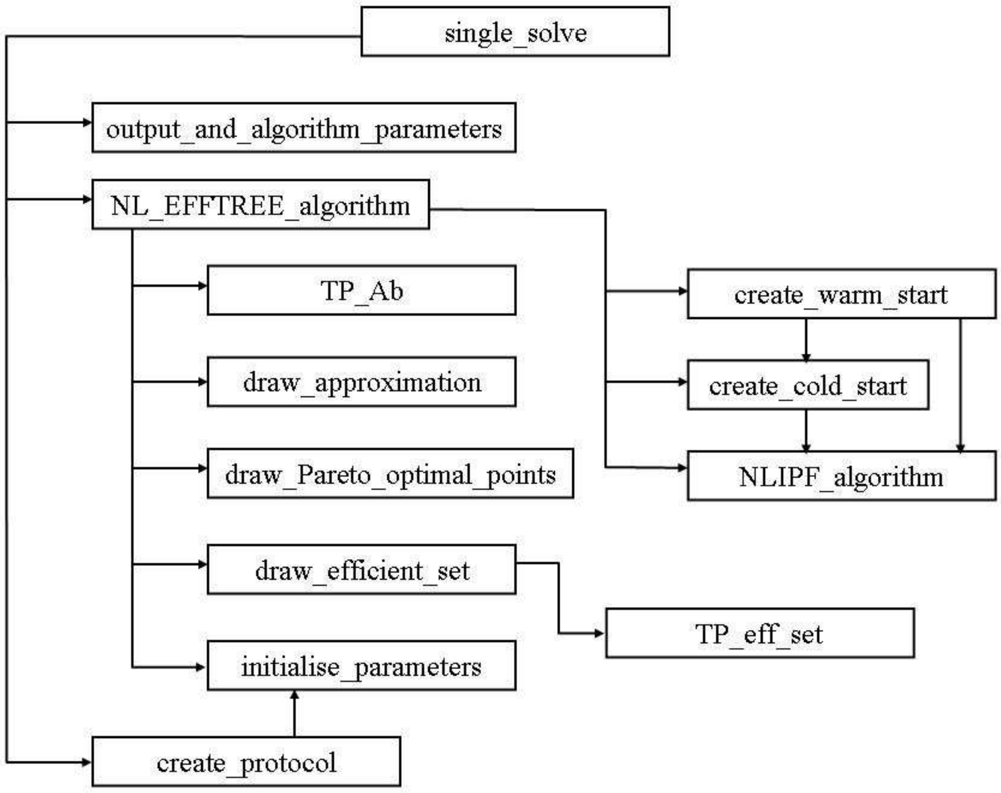


Figure 6.2: Diagram illustrating the operations of the computer code (1).

functions. Conveniently, we aimed to integrate the main task of each M-file into its corresponding name as well.

To ease the process of calibrating the algorithm and observing how different parameter settings impact on the performance of the computer code we decided to store all parameters specifying the algorithm in a central file named `initialise_parameters.m`. Furthermore, the parameters determining the graphical display of the approximations produced by the algorithm as well as the genuine efficient set (if known) can be altered in this M-file.

Another file that gives the user the opportunity to modify the algorithm is entitled `output_and_algorithm_parameters.m`. In particular, different output options can be selected in this M-file, for instance one can opt for a progress update in the command

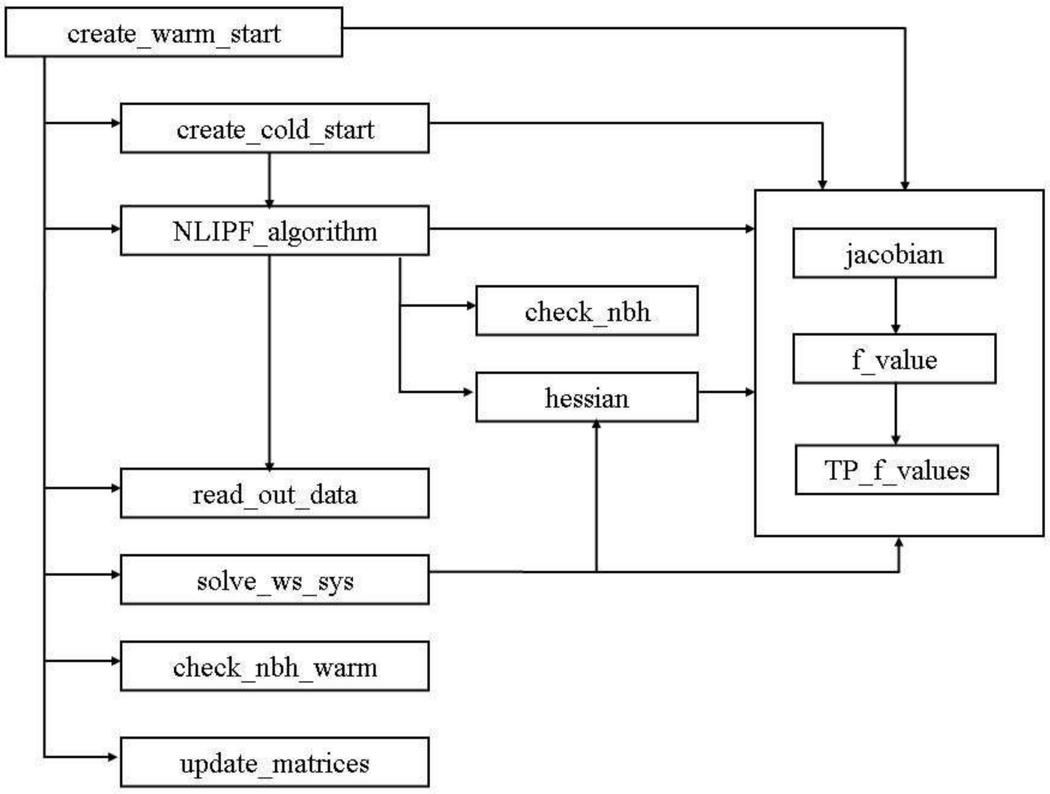


Figure 6.3: Diagram illustrating the operations of the computer code (2).

window during the course of the algorithm as well as a graphical display of the current approximation of the efficient line. Furthermore, one can request that the computer code generates a protocol at the end of the algorithm listing the employed parameters as well as structuring the results and providing statistics about the computation. Moreover, certain features of the algorithm itself can be set in this file. For instance, one option allows the user to prohibit cold starts completely. Another parameter enables the algorithm to execute further iterations of the algorithm NLIPF for individual iterates which greatly violate the duality gap condition in order to smoothen the efficient line. Finally, there is also an opportunity to compute the optimal solution for just a single problem instance with a specific weighting parameter $\omega_{initial}$, which is especially useful if one wants to fine-tune the parameter setting for the algorithm NLIPF.

| Routine | Description |
|---|---|
| `single_solve.m` | main routine for the computation of a test problem |
| `initialise_parameters.m` | setting of all parameters for the algorithms NL_EFFTREE and NLIPF |
| `output_and_algorithm _parameters.m` | setting of output options and algorithm features |
| `NL_EFFTREE_algorithm.m` | main routine for the algorithm NL_EFFTREE |
| `create_cold_start.m` | generation of a cold start<br>- initialsation of the starting point<br>- computation of a cold start if warm-start search fails |
| `create_warm_start.m` | warm-start search for new iterates |
| `NLIPF_algorithm.m` | execution of $i_k$ iterations of the algorithm NLIPF |

Table 6.2: List of major routines

### 6.3.1 Data structure and major routines

Before we can investigate the specific aspects of different sections of the algorithm NL_EFFTREE we have to examine the underlying data structure that is exploited by all corresponding MATLAB routines. Extensive information about every single iterate which is essential during the course of the algorithm is stored in three different arrays, namely `data_matrix`, `NB_matrix` and `NB_distance_matrix`.

The algorithm NLIPF as well as the algorithm NL_EFFTREE requires detailed information about the previous iterates in order to update them or to compute new warm-start iterates respectively. Consequently, it proves to be crucial that the parameter values for each iterate are stored in a concise way so that they can easily be re-used. For that purpose we define the array `data_matrix` which keeps track of the data linked to the individual iterates and whose exact list of entries saved for each iterate can be found in Table 6.3.

Unfortunately, this large amount of data that needs to be saved and read out con-

| Argument | Description |
|---|---|
| $\omega$ | weighting parameter |
| $\mu$ | duality gap |
| $f_1$ | function value of the first objective function |
| $f_2$ | function value of the second objective function |
| $x$ | $x$ component of the iterate |
| $\lambda$ | $\lambda$ component of the iterate |
| $s$ | $s$ component of the iterate |
| $r_b$ | residual $r_b$ |
| $r_c$ | residual $r_c$ |
| $r_{xs}$ | residual $r_{xs}$ |
| $\sigma$ | centering parameter |
| $r_b^0$ | residual $r_b$ of the starting point |
| $r_c^0$ | residual $r_c$ of the starting point |
| $\mu_0$ | duality gap of the starting point |
| $r_{bc}/\mu_0\ \beta$ | norm of the residuals of the starting point divided by the initial duality gap multiplied by the neighbourhood parameter $\beta$ |
| $\beta$ | neighbourhood parameter |
| $\gamma$ | neighbourhood parameter |
| *depth* | number of NLIPF iterations computed |

Table 6.3: List of entries saved for each iterate in the array data_matrix

stantly does not favour the efficiency of the computer code. However, the advantage of storing such vast volumes of data is that one gets an extensive insight in the procedure of the algorithm NL_EFFTREE, since for instance even single trace of iterates could be extracted if desired.

With respect to the warm-start search it is important to introduce two further matrices which provide us with necessary information that is required. Firstly, the matrix `NB_matrix` registers all neighbouring instances of the current efficient line and marks the entries $(i, j)$ and $(j, i)$, where $i$ and $j$ are the columns of the array `data_matrix` in which the corresponding iterates are saved. Secondly, the array `NB_distance_matrix` marks all indices pairs $(i, j)$ whose corresponding iterates $i$ and $j$ do not satisfy the required maximal distance between their respective image points yet. In addition, we want to note that while having dimensions $n \times n$, both neighbouring matrices are extremely sparsely assigned with $2n - 2$ entries each and consequently are defined as SPARSE matrices in the MATLAB environment. To clarify the concept of the neighbouring matrices we present the following example which should further highlight the reason for employing the SPARSE format for these matrices.

**Example 6.1** *Suppose our current approximation of the efficient set consists of five iterates with the following properties summarised in the subsequent table.*

| *column in* `data_matrix` | *3* | | *5* | | *1* | | *4* | | *2* |
|---|---|---|---|---|---|---|---|---|---|
| *weighting parameter* $\omega$ | *1* | | *0.75* | | *0.5* | | *0.25* | | *0* |
| *distance between image points* | | $> \delta$ | | $\leqslant \delta$ | | $> \delta$ | | $\leqslant \delta$ | |

*Then the matrix* `NB_matrix` *detailing the neighbouring relations is of the form*

$$\texttt{NB\_matrix} = \left(\begin{array}{c|c|c|c|c} & & & 1 & 1 \\ \hline & & & 1 & \\ \hline & & & & 1 \\ \hline 1 & 1 & & & \\ \hline 1 & & 1 & & \end{array}\right)$$

*Furthermore, the matrix* `NB_distance_matrix` *specifying the distances between neighbouring iterates is of the form*

$$\texttt{NB\_distance\_matrix} = \left(\begin{array}{c|c|c|c|c} & & & 1 & 0 \\ \hline & & & 0 & \\ \hline & & & & 1 \\ \hline 1 & 0 & & & \\ \hline 0 & & 1 & & \end{array}\right)$$

Besides the constraint matrix $A$ and the constraint vector $b$ as well as the identity matrix $Id \in \mathbb{R}^{n\times n}$ and the unity vector $e = (1,\ldots,1)^T \in \mathbb{R}^n$ these three matrices are the only globally defined variables in the MATLAB computer code for the algorithm NL_EFFTREE.

### 6.3.2 Implementation of the algorithm NLIPF

In this section we investigate the parameters that are utilised by the solution method NLIPF, which is implemented as the routine `NLIPF_algorithm.m` in the computer code. We begin our examination with the parameters of the central path neighbourhood $\mathcal{N}_{-\infty}(\gamma,\beta)$ and conclude by analysing the number of iterations of NLIPF we want to compute during each loop of the overall algorithm NL_EFFTREE. The latter is precisely providing the link between these two methods.

#### Neighbourhood parameters

With regard to the parameters that define the central path neighbourhood $\mathcal{N}_{-\infty}(\gamma,\beta)$, namely $\beta \geqslant 1$ and $\gamma \in (0,1)$ we revert to the values introduced by Heermann [18]. The proposed choice of

$$\beta := 1.2 \quad \text{and} \quad \gamma := 10^{-4}$$

is also employed in Molz et al. [33] and leads to very good results for all our test problems as well.

#### NLIPF parameters

Concerning the parameter that is regulating the maximum duality gap we choose

$$\varepsilon := \sqrt{\texttt{eps}}$$

where `eps` represents the machine precision with regards to floating point arithmetic. In the particular case of using MATLAB as software platform this value is $\texttt{eps} = 2^{-52}$. The centering parameter $\sigma_k$ is chosen dynamically in every loop via the heuristic

$$\sigma_{k+1} := \left(\frac{\mu_{k+1}}{\mu_k}\right)^3.$$

Hence, if we do not observe a sufficiently large decrease in the duality gap, the algorithm automatically induces a centering step that promises substantial progress in the next iteration. Otherwise we are aiming to reduce $\mu_k$ significantly via suitably small values for $\sigma_k$.

**Number of iterations of algorithm NLIPF per loop**

Regarding the parameters $i_0$ and $i_k$ ($k \geqslant 1$) which specify the number of iterations of the algorithm NLIPF during every loop of NL_EFFTREE we follow the proposition of Heermann [18] and set the parameters to

$$i_0 := 5 \quad \text{and} \quad i_k := 1 \quad (k \geqslant 1).$$

At this point we want to highlight that the parameter $i_0$ which regulates the number of initial iterations of the algorithm NLIPF for the starting point is responsible for the proximity to the efficient set of the point from which we initiate the warm-start search. Furthermore, this point is also affected by our choice of $\zeta$ which determines the starting point. For sake of simplicity we stick to the value of $i_0$ suggested by Heermann [18] and only alter the value $\zeta$ to adjust to the individual properties of different test problems.

### 6.3.3 Implementation of the warm-start strategy

In this subsection we take a closer look at the mechanism of the warm-start strategy and in particular, investigate the determination of the perturbation strength factor $\Delta\omega$. This parameter is utilised to generate a warm-start iterate with weighting parameter $\tilde{\omega}$ between an already computed iterate associated to instance $\omega$ and the corresponding neighbouring iterate with weighting parameter $\widehat{\omega}$. We recapitulate that the weighting parameter for the warm-start iterate is consequently determined by the formula $\tilde{\omega} := \omega + \Delta\omega$. Furthermore, in order to verify the feasibility of the perturbation $\Delta\omega$ we have to check if the vector $(\Delta x, \Delta\lambda, \Delta s)$ satisfying the system of linear equations (6.3) (with $\Delta\omega$ on the right-hand side) leads to an admissible warm-start iterate $(\tilde{x}, \tilde{\lambda}, \tilde{s})$ in a new neighbourhood $\mathcal{N}_{-\infty}(\gamma_{\tilde{\omega}}, \beta_{\tilde{\omega}})$.

Thus, we are interested in determining a feasible perturbation vector in as few computational steps as possible, or to initiate a cold start if the warm-start search proves to be unsuccessful. Heermann [18] presented a technique exploiting the concept of the Neumann series to achieve this, however this approach proved to be inferior to the backtracking method which we also utilise in the NL_EFFTREE algorithm. In the following we explain this concept in further details.

**Backtracking strategy to determine the perturbation strength factor $\Delta\omega$**

The approach of our choice, the backtracking strategy, represents a commonly used and straight-forward method in solving problems of that category. Effectively, for $\varepsilon > 0$ we set the parameter

$$\Delta\omega := (0.5 - \varepsilon)(\widehat{\omega} - \omega)$$

and hence, generate a sensible choice of $\Delta\omega$ for both cases $\tilde{\omega} > \omega$ and $\tilde{\omega} < \omega$. If the iterate we aim to perturb is at the boundary of the current approximation, in particular this is the case for the initial iterate, we set the parameter

$$\Delta\omega := \omega - \omega_{min} \quad \text{or} \quad \Delta\omega := \omega_{max} - \omega$$

depending at which end of the approximation we want to initiate a warm start. Subsequently, in case we cannot obtain a feasible perturbation from system (6.3) we reduce the value of $\Delta\omega$ by multiplying it with a constant $\Delta\omega_{multiplier} \in (0, 1)$ and re-evaluate this system of linear equations again. This procedure is repeated until a suitable perturbation is attained or in the worst case $\Delta\omega$ is rejected for being too small and instead a cold start is carried out with the weighting parameter

$$\tilde{\omega} := \omega + 0.4(\widehat{\omega} - \omega).$$

Regarding the calibration of the multiplying factor, it is apparent that greater values for $\Delta\omega_{multiplier}$ will lead to better approximations of the maximally possible perturbation; however this may be linked to longer running times of the computer code. Conversely, smaller values of $\Delta\omega_{multiplier}$ may decrease the running time; yet one will have to put up with the fact that the algorithm may miss out on some feasible warm-start iterates in certain occasions. In the extreme case if the multiplying factor is chosen (far) too small it may happen that especially in the first few iterations maximal perturbations will not be achieved. Consequences of this effect can be that only a subset of the weighting parameter interval is exploited or that the efficient set features 'holes' which may not be 'closed' during the maximal number of loops of the NL_EFFTREE algorithm. Heermann [18] performed comprehensive testing with regard to this issue and indeed, we obtain very satisfying results for our data with his suggested choice of $\Delta\omega_{multiplier} := 0.8$ for the multiplying factor.

We want to remark that the perturbation strengths $\Delta\omega$ for an iterate $\omega$ do not necessarily have to be identical for $\widehat{\omega} > \omega$ and $\widehat{\omega} < \omega$. For instance, it can happen that we are able to achieve the maximum perturbation strength in one direction, while it takes a few iterations of the backtracking algorithm to determine a feasible warm-start iterate in the other direction.

To exemplify this phenomenon we analyse the first three iterates of test problem F12_G1_H1_10 which will be presented in more details in the next chapter. Starting with the initial weighting parameter $\omega = 0.5$ the maximum perturbation strengths for the initial warm-start search are

$$\Delta\omega = \omega_{max} - \omega = \omega - \omega_{min} = 0.5$$

for $\omega_{max} = 1$ and $\omega_{min} = 0$, respectively. Indeed, we reach a feasible warm-start iterate for $\omega_r = 1$ and hence the perturbation strength in this case is $\Delta\omega = 0.5$. However, with respect to $\omega_l$ we have to go through a few iterations of the backtracking algorithm to obtain a feasible iterate and eventually determine one for $\omega_l = 0.224$, which corresponds to a perturbation strength of $\Delta\omega = 0.256$. The relationship between the different perturbation strengths for $\omega_l$ and $\omega_r$ for the initial warm start search of problem F12_G1_H1_10 is illustrated in Figure 6.4.

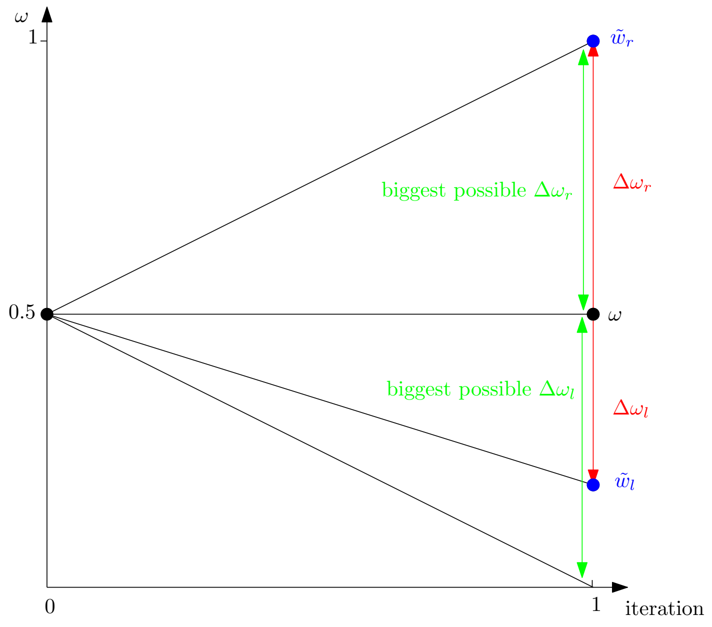


Figure 6.4: The different perturbation strengths for $\omega_l$ and $\omega_r$ for the initial warm start search of problem F12_G1_H1_10.

**Decomposition of the search interval for weighting parameters**

In order to exert some influence on the distribution of the weighting parameters $\omega$ in the search interval during the iteration process of the algorithm NL_EFFTREE we introduce the parameter $\Delta\omega_{minimal}$ which acts as a minimal distance between different weights. More precisely, before we initiate the warm-start search between two neighbouring iterates with weighting parameters $\omega$ and $\widehat{\omega}$ which do not satisfy the duality gap condition and whose distance in the image space is still greater than $\delta$ we verify that the additional condition

$$|\omega - \widehat{\omega}| \geqslant \Delta\omega_{minimal}$$

is fulfilled. If this requirement is satisfied the algorithm proceeds with the warm-start analysis; if this is not the case we refrain from this procedure. The idea behind this approach is to avoid the computation of an excessive number of iterates between $\omega$ and $\widehat{\omega}$ and thus squander valuable computing time on a very small interval of the search space at a premature stage.

In case a desired number of points is provided the decomposition parameter should correlate to this value. A sensible choice for $\Delta\omega_{minimal}$ can be derived from the following formula

$$\Delta\omega_{minimal} := \frac{\omega_{max} - \omega_{min}}{number\ of\ points\ desired}$$

which effectively decomposes the search interval in as many components as points that are desired. The advantage of this strategy is that the running time of the computer programme can be improved since many redundant warm-starts are prevented.

### 6.3.4 Implementation of the algorithm NL_EFFTREE

To conclude our analysis of the MATLAB implementation of the algorithm NL_EFFTREE we present the remaining aspects of the computer code in this last subsection and provide a table with a complete list of all parameter values that were utilised in the MATLAB routines at the very end.

**The starting point parameter $\zeta$**

A short parameter study showed that a choice of

$$\zeta = 10x_{max},$$

where $x_{max}$ is the maximal value any element of the solution vector $x$ can attain, provided the best results when compared to alternatives of $\zeta = 2x_{max}$ and $\zeta = 100x_{max}$.

**Abort of the iteration process**

The algorithm NL_EFFTREE terminates the iteration process if the duality gaps of all iterates are reduced sufficiently to a value below $\varepsilon$ and if the distances between all neighbouring image points do not violate the maximal distance condition. In addition, we employ a loop counter and set the maximal number of loops of the algorithm NL_EFFTREE to `loops_max` $= 50$. An immediate consequence of this choice is that the maximum number of points that could be computed during the course of the algorithm is $3^{50}$. However, in practice the actual number of generated solution points is obviously dependent on the choice of $\delta$ which regulates the density of the image points.
To assist the user of the computer code in calibrating a suitable choice for $\delta$ we present a heuristic in the subsequent section that derives a value for $\delta$ from the number of desired points.

**Heuristic for the determination of $\delta$**

An additional characteristic that is incorporated in the computer code is an automatic estimation of $\delta$ after a chosen number of desired points is entered in the algorithm. To activate this feature we have to set the parameter $\delta := 0$ and decide on `number_of_points` $:=$

*number of points desired* in the M-file `initialise_parameters.m`. Consequently, the routine `delta_set.m` will solve both instances for the boundary weighting parameters $\omega_{minimal}$ and $\omega_{maximal}$ and subsequently, approximate the maximal distance between image points via the formula

$$\delta := \frac{\sqrt{2}\|\varphi_k(\omega_{minimal}) - \varphi_k(\omega_{maximal})\|_2}{\textit{number of points desired}}.$$

**Remark 6.4** *At this point, we want to highlight that this approach and in particular the* `delta_set.m` *function stems from a framework that deals with continuous efficient lines. In case the efficient set consists of disconnected parts this technique to exploit a maximal distance between image points leads to absurdity, since by definition this efficient set has to have points with a distance* $> 0$ *between them to comply with the property of discontinuity.*

To conclude this section we present Table 6.4 with all parameter values that were employed during the solving process of the different test problems. Solely, the parameter $\zeta$ has to be adjusted to the meet the individual requirements of the different test instances.

| Parameter | Value |
|---|---|
| number of points | 200 |
| loops_max | 50 |
| $\varepsilon$ | $\sqrt{eps}$ |
| $\delta$ | 0.01 |
| $\zeta$ | $10\ x_{max}$ |
| $\omega_{initial}$ | 0.5 |
| $\omega_{min}$ | 0 |
| $\omega_{max}$ | 1 |
| $\theta$ | 0.1 |
| $\Delta\omega_{minimal}$ | 0.001 |
| $\Delta\omega_{multiplier}$ | 0.8 |
| $i_0$ | 5 |
| $i_k$ | 1 |
| $\alpha$ | 0.84 |
| $\sigma_{initial}$ | 0.1 |
| $\beta$ | 1.2 |
| $\gamma$ | $10^{-4}$ |

Table 6.4: List of parameter values

# Chapter 7

# Test Problems

This chapter presents and scrutinises the numerical results which we obtained from the computational implementation of our solution method for a number of test suites. Initially we review the different sources of the test problems that are cited in the literature. Subsequently we categorise the test problems and provide details about their key properties. This information is followed by graphical illustrations of the approximations of the efficient sets that our algorithm determined to deliver a visual insight into the performance of our computer code. Likewise a structured presentation of the data collected during the computations is provided in order to evaluate the operations of our computer programme and to serve as performance measures.

## 7.1 Format of the test problem data

Before we can examine the outcomes of the numerical implementation in details it is important to clarify the format we utilised to formulate the various test problems in MATLAB. We decided to create three files to describe each test case. In particular, each file name consists of the problem name followed by the ending `_f_values.m`, `_Ab.m` or `_eff_set.m`, respectively. Firstly, the file `testproblem_f_values` contains the two objective functions that define the corresponding test problem, while the related constraint matrix $A$ and the constraint vector $b$ are saved under the file `testproblem_Ab`. Secondly, if information about the efficient set is available we create a file labelled `testproblem_eff_set` which is utilised to plot the efficient set that we aim to approximate for the particular test problem.

At this point we want to highlight that this data format appears to be the most convenient and user-friendly way to implement the numerous test problems in a consistent way. However, from a perspective of minimising the running time of the computer programme this set-up seems to be inappropriate since extensive functions calls have a very negative impact on the performance of the MATLAB code. Concluding, we deemed it best to maintain the user-orientated format and to allocate a lower priority to the optimisation of the running time at this stage of the development of the algorithm.

## 7.2 Test suites

With regard to the compilation of the different test problems we consider the test suites developed by Deb [6] and Zitzler et al. [50] as well as the collection assembled by Van Veldhuizen [42], [43]. An excellent overview over the existing test problems in the literature and means to classify these is provided in the paper by Huband [21] in which he also highlights the above mentioned test suites as the most prominent works in this field.

### 7.2.1 Deb's toolkit

The toolkit devised by Kalyanmoy Deb [6] epitomises an excellent methodology for the initial testing stage of an algorithm. Single test problem properties can be chosen independently and hence, this framework provides an ideal opportunity to analyse how the computer code copes with several different problem settings. In particular, the test problems are constructed via the help of three auxiliary functions. Thus, the two objective functions $f_1$ and $f_2$ of the multiobjective optimisation problems that need to be minimised are of the general form (unless otherwise stated)

$$\begin{aligned} f_1(x) &= f_1(x_1, \ldots, x_m) \\ f_2(x) &= g(x_{m+1}, \ldots, x_n) \times h(f_1, g) \\ & x_i \in [0, 1] \end{aligned} \tag{7.1}$$

where the functions $f_1$, $g$ and $h$ are specified to create different problem properties. In his paper Deb suggests 5 alternatives for $f_1$, 5 for $g$ and 4 for $h$, respectively. Due to the inherent binary nature we neglect the fifth proposal for $f_1$ and two choices for function $g$. Consequently we construct 12 test problems of our own based on this toolkit; the precise formulation of which can be found in Appendix B. In addition, we want to point out that the test problems devised using Deb's toolkit are labelled `F1a_Gb_Hc_d` in the subsequent analysis where `a`, `b` and `c` specify the corresponding choices for the functions $f$, $g$ and $h$, respectively and `_d` is an optional ending to denote problem inherent parameters.

### 7.2.2 ZDT test suite

The "immensely popular" (Huband [21]) test suite designed by Zitzler, Deb and Thiele [50] is based upon Deb's toolkit and deduces six benchmark problems for multiobjective optimisation. Consequently these problems show similarities to the ones developed via Deb's toolkit, in particular they are constructed in the same way exploiting framework (7.1). However, the individual objective functions are of higher complexity and also the dimensions of the solution vectors are considerably higher, which qualifies this arrangement of problems to be the "most widely employed suite of benchmark multiobjective problems in the EA [evolutionary algorithms] literature." (Huband [21]). Similar to the situation before, we ignore the fifth test problem in our investigation due to the binary characteristics of this instance. Furthermore, in addition to computing approximations of the efficient sets for the problems suggest by Zitzler et al we increase the dimensions for the first three test problems to examine the performance of our method for larger scaled problems. Hence, we derive 8 test problems of the form (7.1) whose exact formulations are presented in Appendix B. In the following investigation we refer to these problems as `ZDT_a_b` where `a` denotes the particular problem of this test suite and `_b` is an optional parameter that indicates the dimension of the problem instance.

### 7.2.3 Van Veldhuizen's test suite

Based on his research efforts in his PhD Thesis [43] and additional papers (most prominently [42]) the remaining collection of test problems is usually associated with Van Veldhuizen. These are various test problems that emerged in the literature over time and were gathered by Van Veldhuizen. We note that the book by Coello et al. [5] serves as reference point, since it represents the latest version of this assemblage known to the author. Focussing on test problems with two objectives and linear constraints only we concentrate on 14 problem instances which are examined. The precise formulations and parameter settings for these cases can be found in Appendix B. With regards to the denotation of the test problems we follow Van Veldhuizen's labelling and utilise the names of the original authors.

## 7.3 Test problems

For the following examination we divide the considered test problems into three categories depending on their individual characteristics. In particular the three classes derive from the convexity properties of the single objective functions and hence, we separate the problem instances into convex, concave and nonconvex problems. For a test problem to qualify for the convex category both objective functions have to be convex. Similarly both objective functions have to be concave in order for a problem instance to be admitted to the group of concave test problems. Finally if a least one of the objective functions is nonconvex and nonconcave we consider this problem to be a member of the set of nonconvex test problems. Consequently, we analyse 11 convex, 5 concave and 18 nonconvex test problems and discuss the numerical results we obtained for these problem instances in the following subsections.

In each of these sections we initially recapitulate the fundamental properties of the test problems in a table and provide brief comments about specific characteristics of certain test problems as appropriate. In particular we list the number of variables used in these problem instances and scrutinise the differentiability of the individual objective functions. Moreover, we analyse the convexity properties of the objective functions as well as the shape of the efficient curve. This presentation is followed by graphical illustrations of the approximations of the efficient sets which were determined by our computer algorithm. In these figures all solution points that were computed are indicated by red points and Pareto optimal points are highlighted by blue circles. In addition, the actual efficient set for the particular problem is displayed by green points.

**Remark 7.1** *Due to the numerical nature of the approximations of the efficient sets determined by our computer algorithm the detection of Pareto optimal points proves to be delicate. Often certain points are strictly speaking dominated by other solutions, however their function values only differ marginally. Particularly since the precision of these function values is limited due to rounding errors, we introduce a new notion of Pareto optimality to account for this circumstance. Hence, we denote a solution* $y^* = f(x^*)$ *$\varepsilon$-Pareto optimal if there exists no other solution* $y = f(x)$ *such that*

$$f_1(x) < f_1(x^*) - \varepsilon \quad \textit{and} \quad f_2(x) \leqslant f_2(x^*) - \varepsilon$$

*or*

$$f_1(x) \leqslant f_1(x^*) - \varepsilon \quad \textit{and} \quad f_2(x) < f_2(x^*) - \varepsilon.$$

*In other words, we decrease both functions values of the solution under consideration by $\varepsilon$ before we compare them to the function values of all other solutions in the process of determining Pareto optimality.*
*In the subsequent analysis we opt for $\varepsilon = \sqrt{\texttt{eps}}$, where* `eps` *stands for the machine precision with regards to floating point arithmetic.*

Subsequently we list the key data from the computations in a further table to provide an insight into the operations of our numerical implementation. In particular we state the elapsed time to produce the approximation as well as the number of solution points that are computed and the percentage of $\varepsilon$-Pareto optimal points. These $\varepsilon$-Pareto optimal points are determined during a post processing step in which all computed solution points are compared with each other and the $\varepsilon$-Pareto optimal points are selected. This information is complemented by the number of systems of linear equations that were solved per point, the number of gradient evaluations per point and the number of function evaluations per point to provide performance measures. Finally, short comments assessing the outcomes of the numerical implementations and remarks indicating potentials for improvements of the computer code round off our analysis of the test problems.

### 7.3.1 Convex test problems

As mentioned in the introductory section above we proceed by outlining the key features of the convex problems in Table 7.1 and refer the interested reader to the original papers for a more in-depth analysis. At this point we want to highlight that several objective functions are not continuously differentiable on the entirety of the $\mathbb{R}^n$. However, the points of non-differentiability are generally outside the feasible region; we specify these conditions in columns three and four of Table 7.1. Additionally, we note that the problem `Kita` is the only one featuring genuine side constraints in addition to the typical interval constraints for the variables.

| Problem | variables | $f_1$ | $f_2$ | $f_1$ | $f_2$ | Pareto curve |
|---|---|---|---|---|---|---|
| F11_G1_H1 | 2 | $C^\infty$ | $C^\infty([0,10]^2)$ | lin. | convex | convex |
| F12_G1_H1_10 | 11 | $C^\infty$ | $C^\infty([0,10]^2)$ | lin. | convex | convex |
| F11_G2_H1_2 | 2 | $C^\infty$ | $C^\infty([0,10]^2)$ | lin. | convex | convex |
| F11_G1_H1b_05 | 2 | $C^\infty$ | $C^\infty([0,10]^2)$ | lin. | convex | convex |
| ZDT_1_30 | 30 | $C^\infty$ | $C^\infty((0,10]^{30})$ | lin. | convex | convex |
| ZDT_1_100 | 100 | $C^\infty$ | $C^\infty((0,10]^{100})$ | lin. | convex | convex |
| Binh1 | 2 | $C^\infty$ | $C^\infty$ | quad. | quad. | convex |
| Laumanns | 2 | $C^\infty$ | $C^\infty$ | quad. | quad. | convex |
| Rendon2 | 2 | $C^\infty$ | $C^\infty$ | lin. | quad. | convex |
| Schaffer | 1 | $C^\infty$ | $C^\infty$ | quad. | quad. | convex |
| Kita | 2 | $C^\infty$ | $C^\infty$ | quad. | lin. | convex |

Table 7.1: List of convex test problems. Explanation of the columns: 1 - name of the test problem, 2 - number of variables of the test problem, 3 & 4 - differentiability of the objective functions, 4 & 5 - convexity properties of the objective functions, 6 - convexity property of the efficient set. (lin. = linear, quad. = quadratic)

The performance, that is the quality of the approximations of the efficient set, determined by our solution method for the class of convex functions is impressive as can be seen in all Figures 7.1 to 7.11.

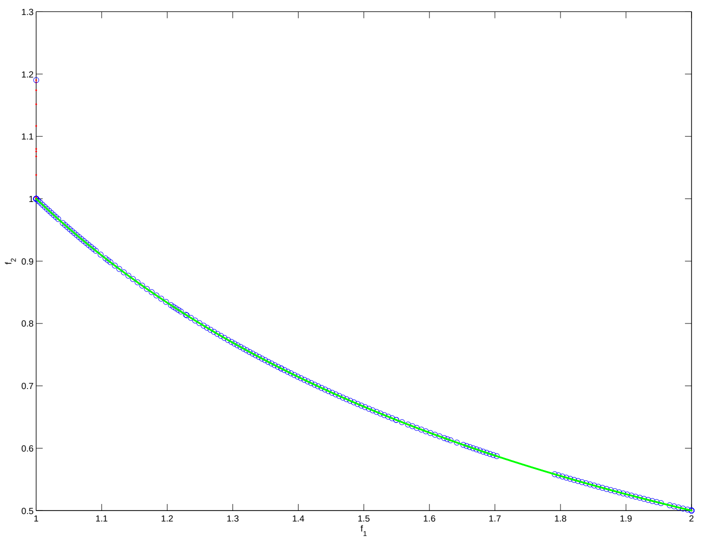


Figure 7.1: Problem F11_G1_H1

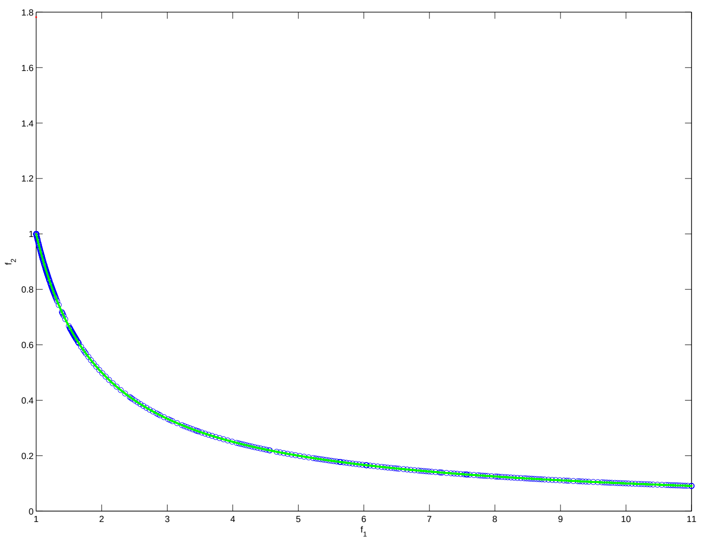


Figure 7.2: Problem F12_G1_H1_10

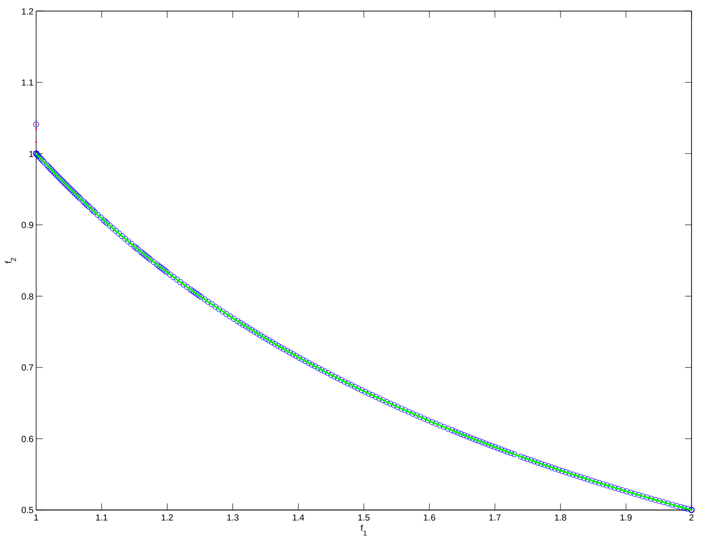


Figure 7.3: Problem F11_G2_H1_2

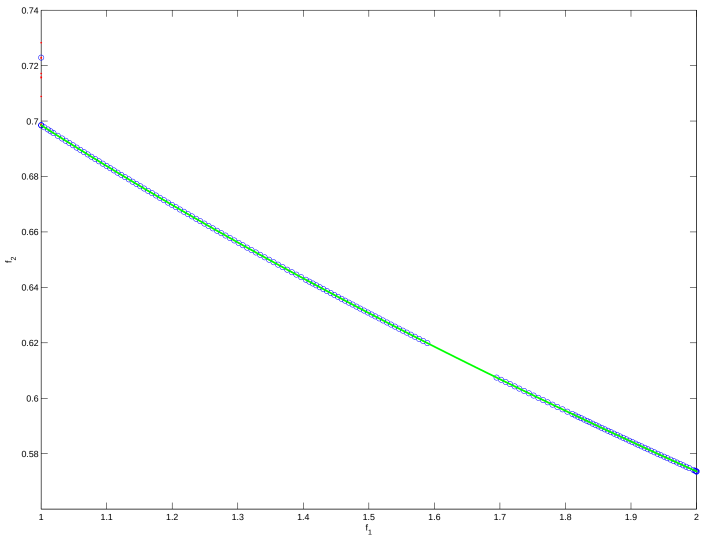


Figure 7.4: Problem F11_G1_H1b_05

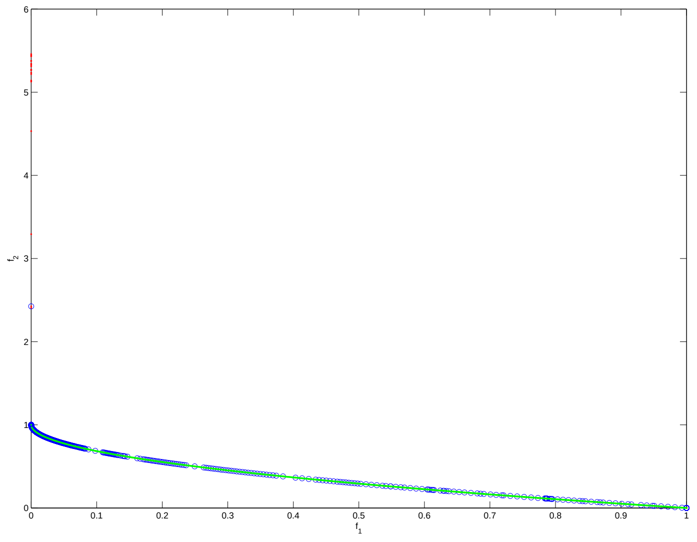


Figure 7.5: Problem ZDT_1_30

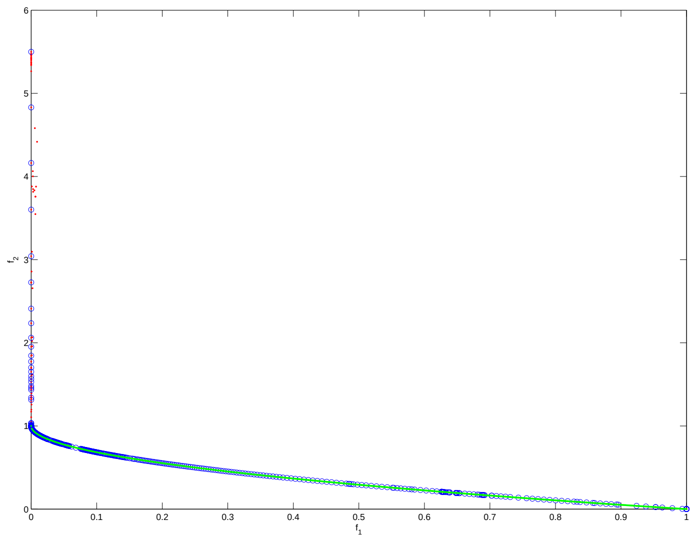


Figure 7.6: Problem ZDT_1_100

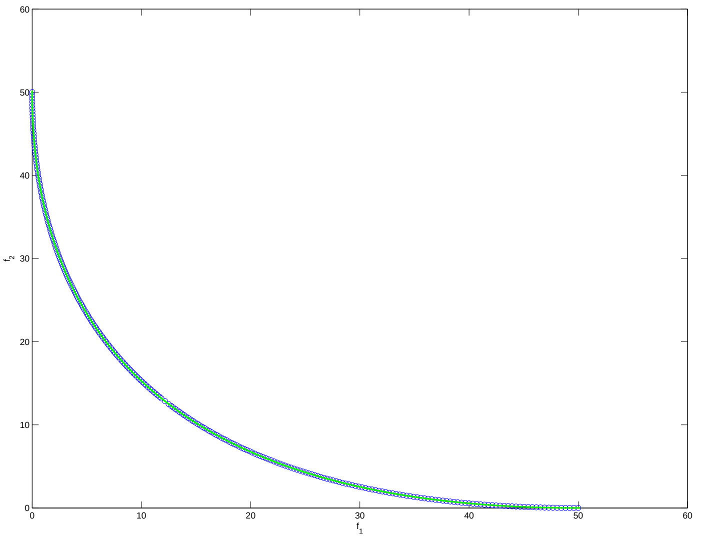


Figure 7.7: Problem Binh1

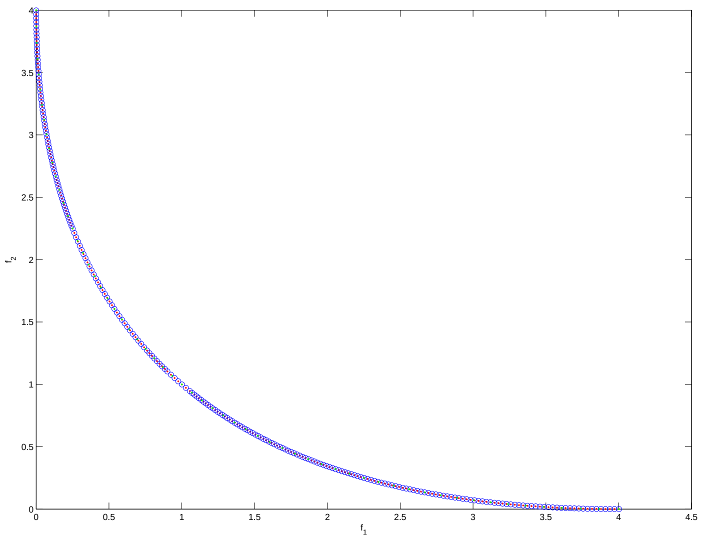


Figure 7.8: Problem Laumanns

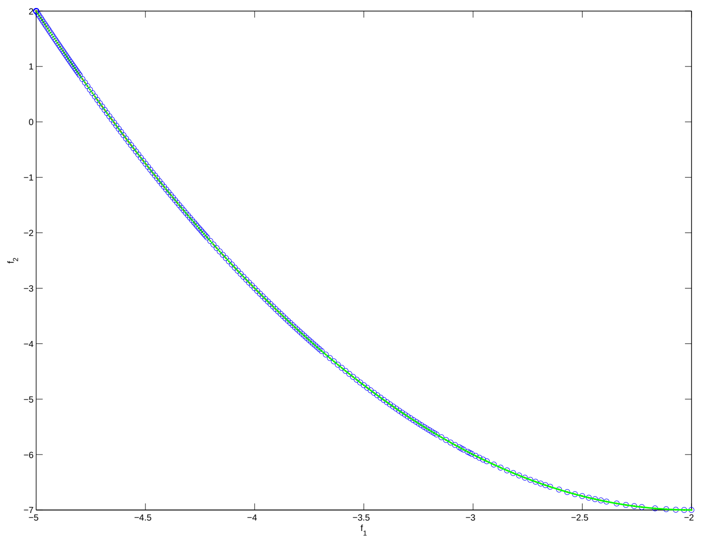


Figure 7.9: Problem Rendon2

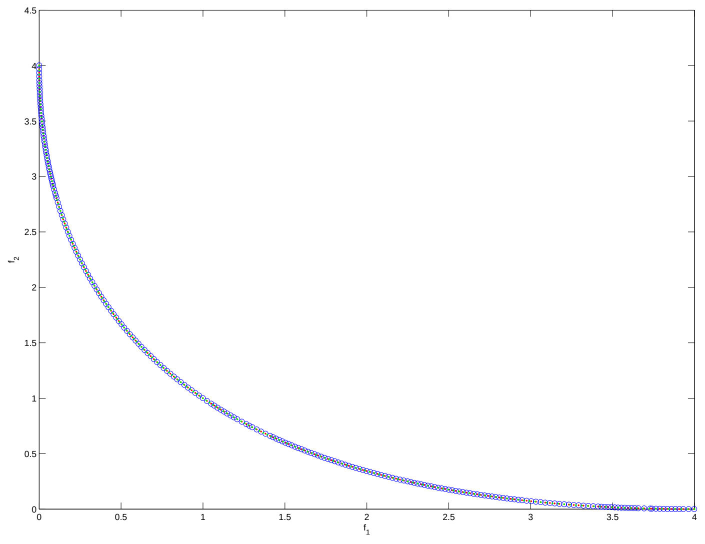


Figure 7.10: Problem Schaffer

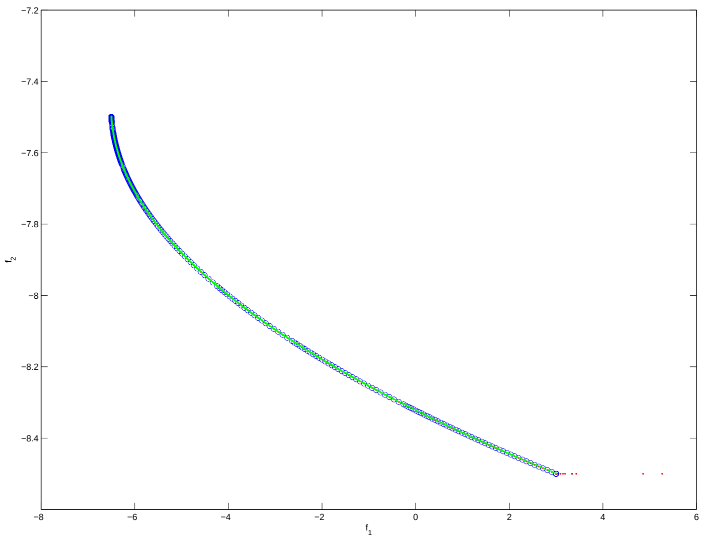


Figure 7.11: Problem Kita

Table 7.2 structures the results we obtain for this set of test problems and presents the performance measures needed to analyse the numerical implementation of the solution algorithm. To sum up the outcomes for this group of test problems we conclude that our solution methods works very well for convex optimisation problems. In particular practically every solution point that we compute coincides with the actual efficient set and is $\varepsilon$-Pareto optimal. Furthermore, we observe that the behaviour of the algorithm remains similar if we scale up the number of variables. For example, the number of linear systems solved per point and the number of function evaluations per point for problems `F11_G1_H1` and `F12_G1_H1_10` as well as for `ZDT_1_30` and `ZDT_1_100` stay virtually the same. In addition, we want to point out that for some test instances the algorithm determines optimal approximations before the maximal number of iterations is reached. For instance, the computer code terminates after 32 iterations for problem `F11_G1_H1b_05`, 9 for `Binh`, 9 for `Laumanns` and 19 for `Rendon2`, respectively. Moreover, we remark that we employ COELLO's formulation of the problem `Schaffer` and thus, consider the interval constraints $x \in [0, 2 \cdot 10^5]$. Consequently our starting point is of the order $2 \cdot 10^6$, which explains the high numbers of systems of linear equations that need to be solved for each solution point, since the initial point is very far away from the actual solution. Finally, we observe that the number of gradient evaluations per point is considerably high for all test problems. This indicates that the line search routine of the algorithm NLIPF shows some room for improvement.

### 7.3.2 Concave test problems

Similar to the previous section we commence by recapitulating the most prominent characteristics of the concave test problems in Table 7.3 and note that more details can be obtained from the original sources. At this point we want to hightlight that in particular the Pareto curve is concave and hence we utilised a squared objective functions approach

| Problem | time (sec) | points | $\varepsilon$-Pareto points (%) | SOLE / pt | Jacobian / pt | Function eval. / pt |
|---|---|---|---|---|---|---|
| F11_G1_H1 | 6.19 | 253 | 100.00 | 7.02 | 171.65 | 5.67 |
| F12_G1_H1_10 | 87.14 | 521 | 97.89 | 7.92 | 244.26 | 7.64 |
| F11_G2_H1_2 | 18.11 | 391 | 97.95 | 12.50 | 346.04 | 12.19 |
| F11_G1_H1b_05 | 1.64 | 192 | 100.00 | 3.52 | 47.77 | 2.26 |
| ZDT_1_30 | 321.25 | 638 | 94.83 | 9.03 | 251.59 | 8.16 |
| ZDT_1_100 | 2103.80 | 714 | 79.97 | 9.33 | 340.41 | 7.73 |
| Binh1 | 1.88 | 260 | 100.00 | 1.89 | 49.58 | 1.84 |
| Laumanns | 3.28 | 239 | 100.00 | 3.53 | 104.92 | 3.49 |
| Rendon2 | 4.63 | 281 | 100.00 | 4.33 | 121.09 | 4.21 |
| Schaffer | 28.83 | 236 | 100.00 | 15.91 | 1151.81 | 15.80 |
| Kita | 11.05 | 479 | 99.37 | 8.75 | 157.34 | 8.23 |

Table 7.2: Results for convex test problems. Explanation of the columns: 1 - name of the test problem, 2 - running time, 3 - number of solution points, 4 - percentage of $\varepsilon$-Pareto optimal points of the solution points, 5 - number of systems of linear equations solved per solution point, 6 - number of Jacobian vector evaluations per solution point, 7 - number of function evaluations per solution point.

(i.e. $f(x) := \omega f_1^2(x) + (1-\omega) f_2^2(x)$), expect for the problem `Fonseca` which exploits the standard technique, since the standard weighted-sum technique cannot determine intermediate Pareto-optimal solutions by using a weight vector (cf. Deb [6, page 217]).

| Problem | variables | $f_1$ | $f_2$ | $f_1$ | $f_2$ | Pareto curve |
|---|---|---|---|---|---|---|
| F11_G1_H1b_1 | 2 | $C^\infty$ | $C^\infty([0,10]^2)$ | linear | linear | linear |
| F11_G1_H2 | 2 | $C^\infty$ | $C^\infty([0,10]^2)$ | linear | concave | concave |
| ZDT_2_30 | 30 | $C^\infty$ | $C^\infty([0,10]^{30})$ | linear | concave | concave |
| ZDT_2_100 | 100 | $C^\infty$ | $C^\infty([0,10]^{100})$ | linear | concave | concave |
| Fonseca | 2 | $C^\infty$ | $C^\infty$ | concave | concave | concave |

Table 7.3: List of concave test problems. Explanation of the columns: 1 - name of the test problem, 2 - number of variables of the test problem, 3 & 4 - differentiability of the objective functions, 4 & 5 - convexity properties of the objective functions, 6 - convexity property of the efficient set.

The results obtained from our computer code for these concave test problems were equally impressive which is illustrated by the graphical representations of the approximations of the efficient sets in Figure 7.12 to Figure 7.16.

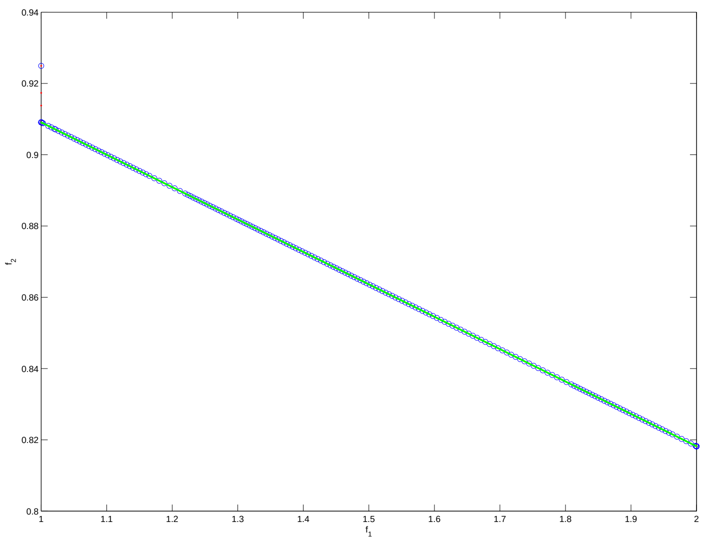


Figure 7.12: Problem F11_G1_H1b_1

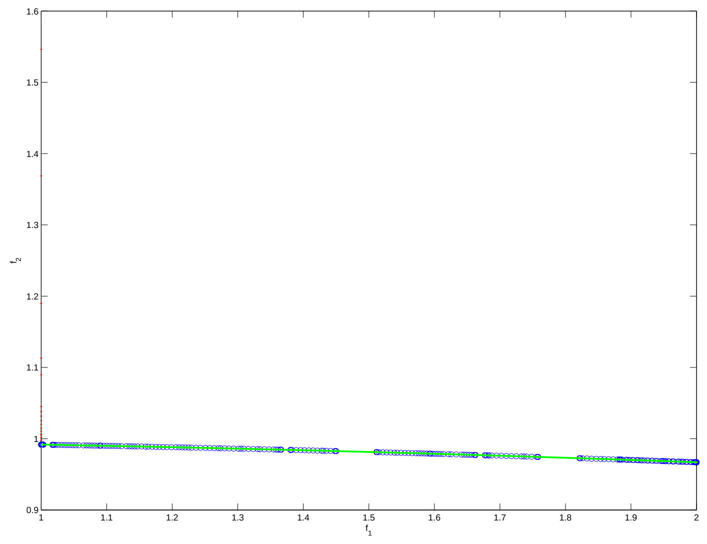


Figure 7.13: Problem F11_G1_H2

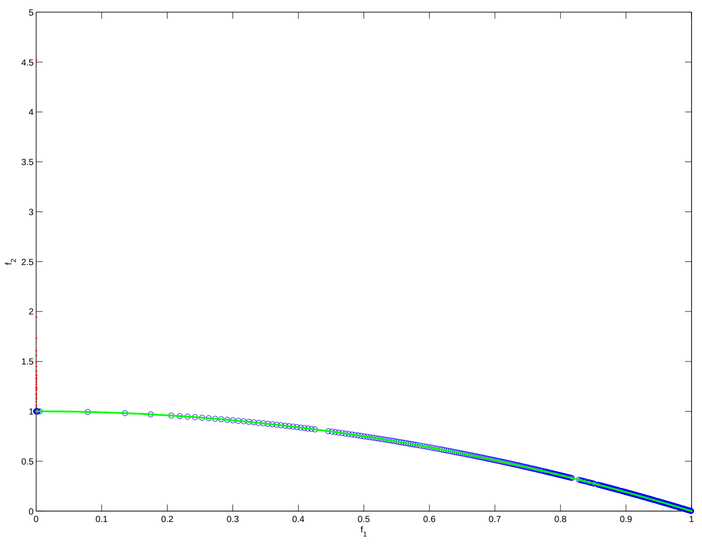


Figure 7.14: Problem ZDT_2_30

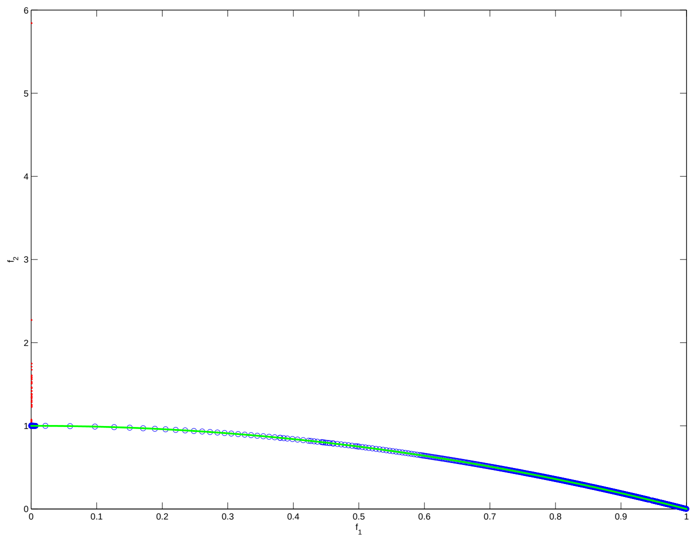


Figure 7.15: Problem ZDT_2_100

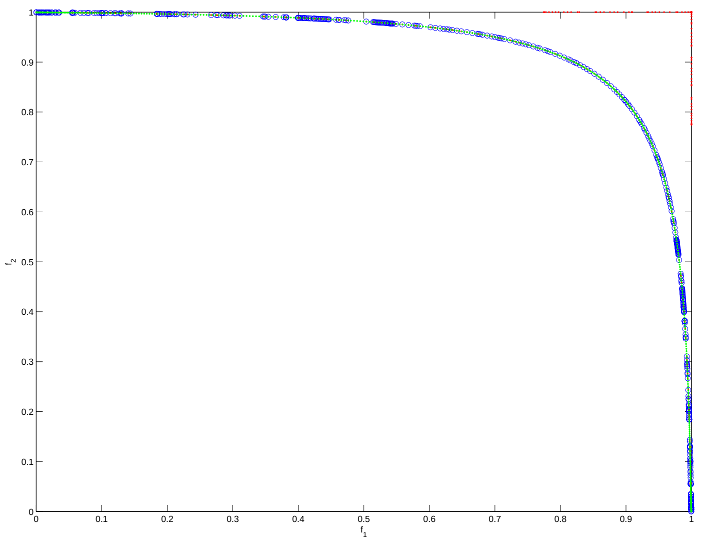


Figure 7.16: Problem Fonseca

The key data from the computations for this group of concave test problems is listed in Table 7.4. Akin to the findings for the convex problems the results we obtained for the concave problem instances are just as formidable. Again, effectively all computed solution points lie on the actual efficient curve and qualify as $\varepsilon$-Pareto optimal. Consequently we deduce that our computer algorithm performs very well for concave optimisation problems too. Furthermore, we notice that the solution method scales up well for larger numbers of variables as well as can be observed for the problems `ZDT_2_30` and `ZDT_2_100`. At last, we see again that the number of gradient evaluations per point is large in all test cases, which suggests that the implementation of the line search procedure is suboptimal.

| Problem | time (sec) | points | $\varepsilon$-Pareto points (%) | SOLE / pt | Jacobian / pt | Function eval. / pt |
|---|---|---|---|---|---|---|
| F11_G1_H1b_1 | 1.58 | 216 | 100.00 | 2.93 | 44.50 | 2.21 |
| F11_G1_H2 | 5.22 | 413 | 100.00 | 4.99 | 82.07 | 3.09 |
| ZDT_2_30 | 269.48 | 586 | 89.25 | 10.15 | 197.59 | 9.44 |
| ZDT_2_100 | 2076.22 | 735 | 80.95 | 11.11 | 216.10 | 10.59 |
| Fonseca | 8.80 | 765 | 80.13 | 3.26 | 77.85 | 2.59 |

Table 7.4: Results for concave test problems. Explanation of the columns: 1 - name of the test problem, 2 - running time, 3 - number of solution points, 4 - percentage of $\varepsilon$-Pareto optimal points of the solution points, 5 - number of systems of linear equations solved per solution point, 6 - number of Jacobian vector evaluations per solution point, 7 - number of function evaluations per solution point.

### 7.3.3 Nonconvex test problems

Finally we draw our attention to the category of test problems that feature at least one nonconvex and nonconcave objective function. Undoubtedly this set of test cases represents the hardest group with several test problems featuring intricate properties that exceed the theoretical abilities of our solution method. For example, these characteristics include highly nonconvex objective functions, non-differentiable points in the feasible set or disconnected Pareto curves. We summarise the most important properties of these problem instances in Tables 7.5 and 7.6 and recommend to consult the original papers for further information.

| Problem | variables | $f_1$ | $f_2$ | $f_1$ | $f_2$ |
|---|---|---|---|---|---|
| F13_G1_H1 | 2 | $C^\infty(\mathbb{R}_+ \times \mathbb{R})$ | $C^\infty([0,10]^2)$ | nonc. | nonc. |
| F14_G1_H1 | 2 | $C^\infty$ | $C^\infty$ | nonc. | nonc. |
| F11_G2_H1_05 | 2 | $C^\infty$ | $C^\infty((0,10]^2)$ | lin. | nonc. |
| F11_G3_H1 | 2 | $C^\infty$ | $C^\infty([0,10]^2)$ | lin. | nonc. |
| F11_G1_H3 | 2 | $C^\infty$ | $C^0$ | lin. | nonc. |
| F11_G1_H4 | 2 | $C^\infty$ | $C^\infty([0,10]^2)$ | lin. | nonc. |
| ZDT_3_30 | 30 | $C^\infty$ | $C^\infty((0,10]^{30})$ | lin. | nonc. |
| ZDT_3_100 | 100 | $C^\infty$ | $C^\infty((0,10]^{100})$ | lin. | nonc. |
| ZDT_4 | 10 | $C^\infty$ | $C^\infty((0,10]^{10})$ | lin. | nonc. |
| ZDT_6 | 10 | $C^\infty$ | $C^\infty((0,10]^{10})$ | nonc. | nonc. |
| Fonseca2 | 3 | $C^\infty$ | $C^\infty$ | nonc. | nonc. |
| Kursawe1 | 3 | $C^\infty(\mathbb{R}\backslash\{(5,5,5)\})$ | $C^0$ | nonc. | nonc. |
| Lis | 2 | $C^\infty(\mathbb{R}\backslash\{(5,5)\})$ | $C^\infty(\mathbb{R}\backslash\{(5.5,5.5)\})$ | nonc. | nonc. |
| Murata | 2 | $C^\infty(\mathbb{R}_+ \times \mathbb{R})$ | $C^\infty$ | concave | nonc. |
| Poloni | 2 | $C^\infty$ | $C^\infty$ | nonc. | quad. |
| Quagliarella | 16 | $C^0$ | $C^0$ | nonc. | nonc. |
| Rendon | 2 | $C^\infty$ | $C^\infty$ | nonc. | quad. |
| Schaffer2 | 1 | $C^0$ | $C^\infty$ | nonc. | quad. |

Table 7.5: List of nonconvex test problems. Explanation of the columns: 1 - name of the test problem, 2 - number of variables of the test problem, 3 & 4 - differentiability of the objective functions, 4 & 5 - convexity properties of the objective functions. (lin. = linear, quad. = quadratic, nonc. = nonconvex)

| Problem | Pareto curve |
|---|---|
| F13_G1_H1 | convex |
| F14_G1_H1 | convex |
| F11_G2_H1_05 | convex |
| F11_G3_H1 | convex |
| F11_G1_H3 | convex (local optima concave) |
| F11_G1_H4 | 4 noncontiguous parts |
| ZDT_3_30 | 5 noncontiguous parts |
| ZDT_3_100 | 5 noncontiguous parts |
| ZDT_4 | convex |
| ZDT_6 | concave |
| Fonseca2 | concave |
| Kursawe1 | 3 noncontiguous parts |
| Lis | concave |
| Murata | concave |
| Poloni | 2 noncontiguous parts |
| Quagliarella | concave |
| Rendon | convex |
| Schaffer2 | 2 noncontiguous parts |

Table 7.6: List of nonconvex test problems. Explanation of the columns: 1 - name of the test problem, 2 - convexity property of the efficient set.

Graphical displays of the approximations of the efficient sets that our solution algorithm determined for these test problems can been seen in Figure 7.17 to Figure 7.40.

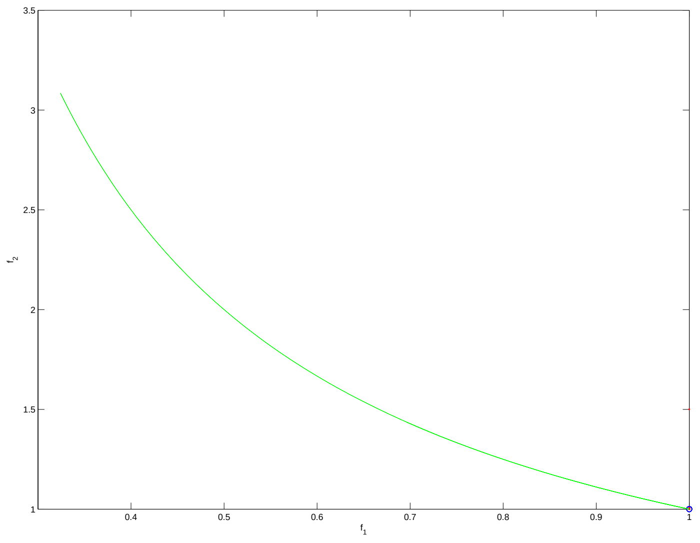


Figure 7.17: Problem F13_G1_H1

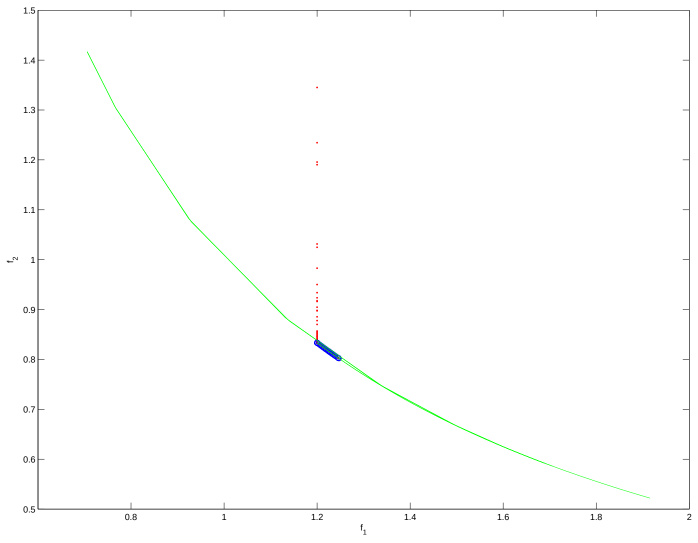


Figure 7.18: Problem F14_G1_H1

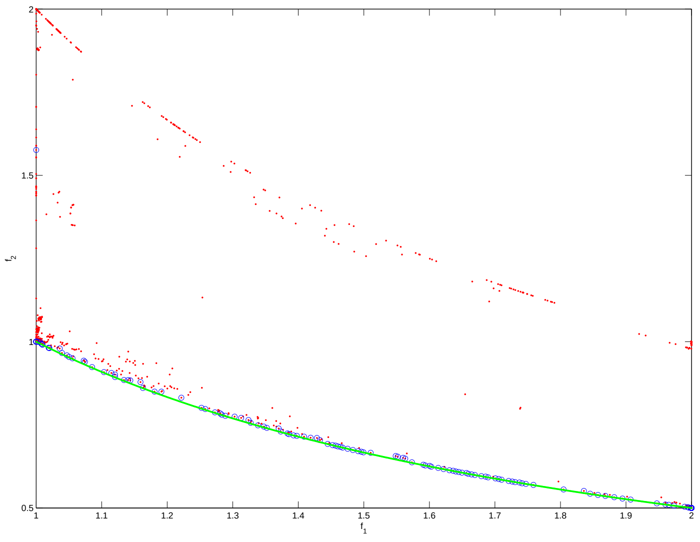


Figure 7.19: Problem F11_G2_H1_05

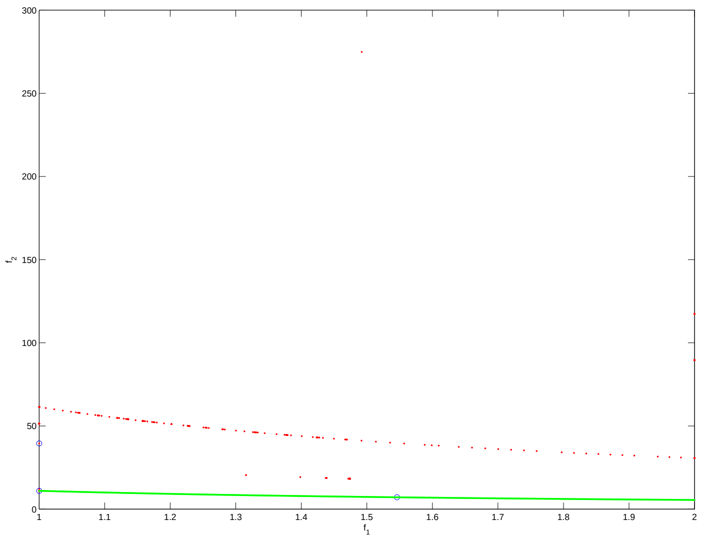


Figure 7.20: Problem F11_G3_H1

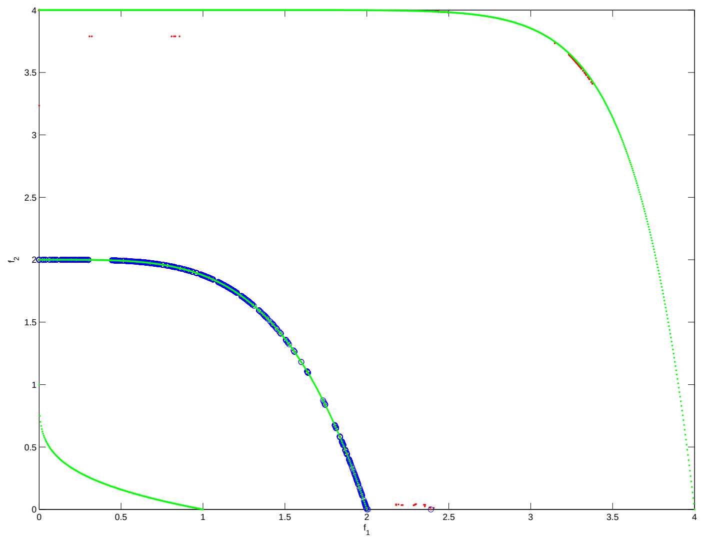


Figure 7.21: Problem F11_G1_H3

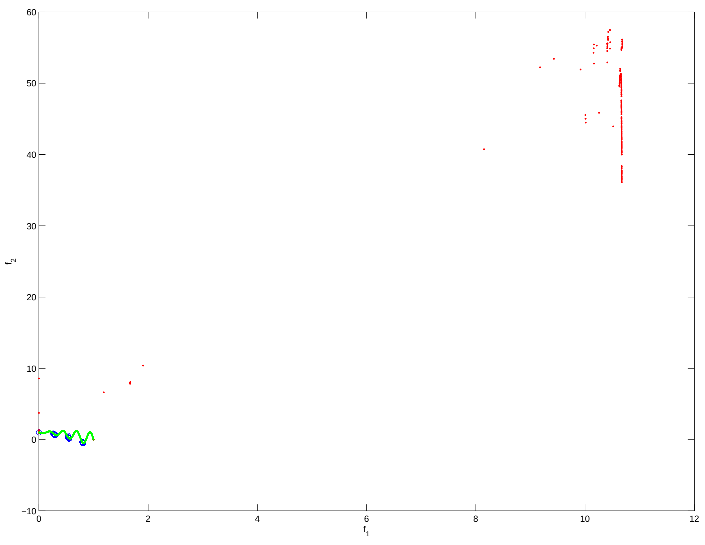


Figure 7.22: Problem F11_G1_H4

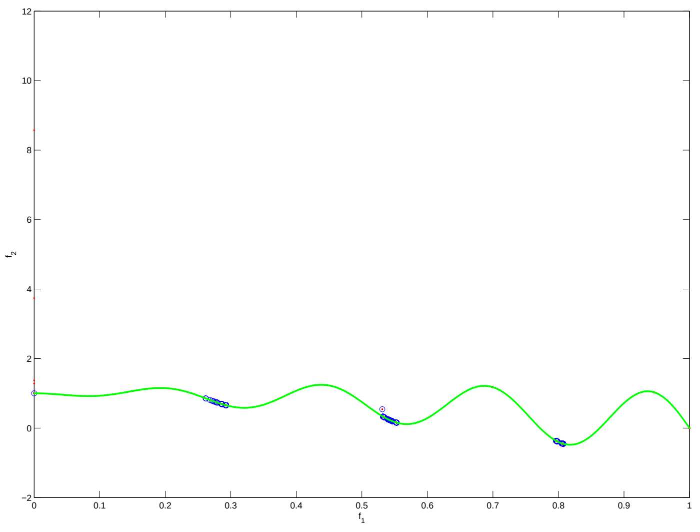


Figure 7.23: Problem F11_G1_H4 (zoomed in)

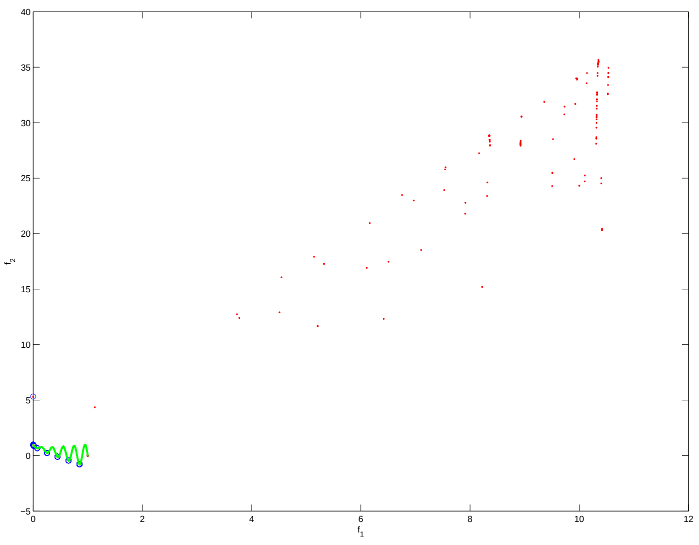


Figure 7.24: Problem ZDT_3_30

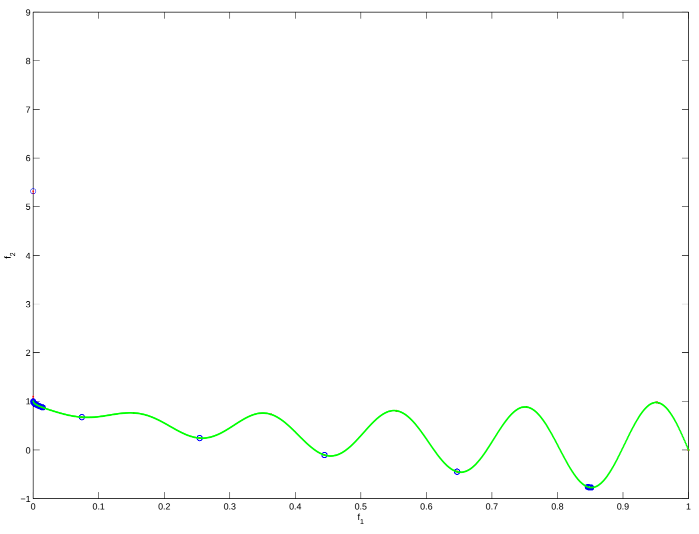


Figure 7.25: Problem ZDT_3_30 (zoomed in)

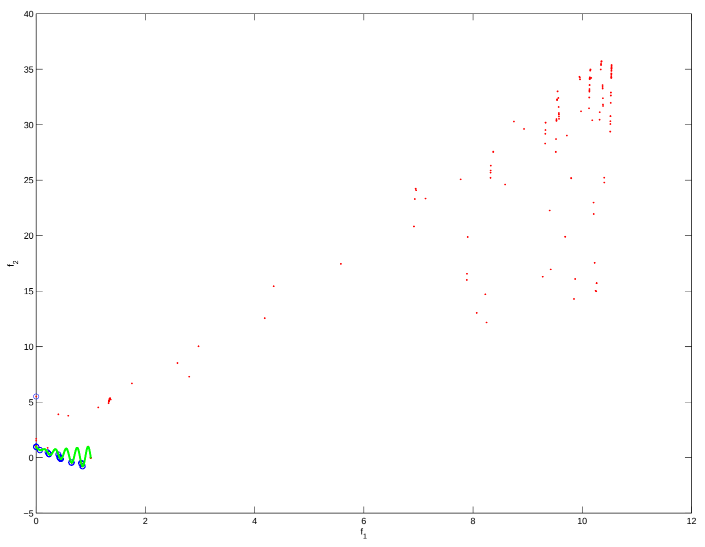


Figure 7.26: Problem ZDT_3_100

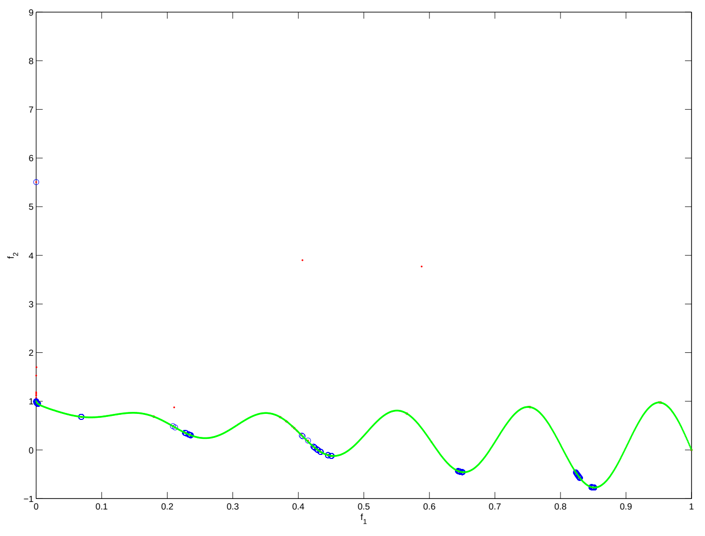


Figure 7.27: Problem ZDT_3_100 (zoomed in)

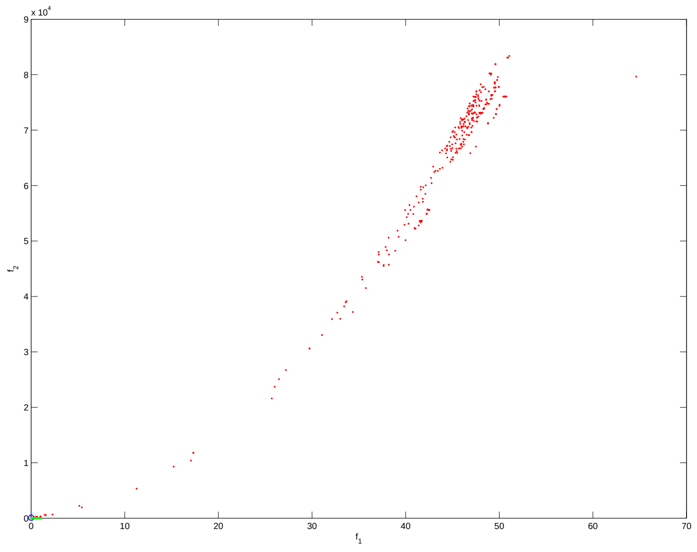


Figure 7.28: Problem ZDT_4

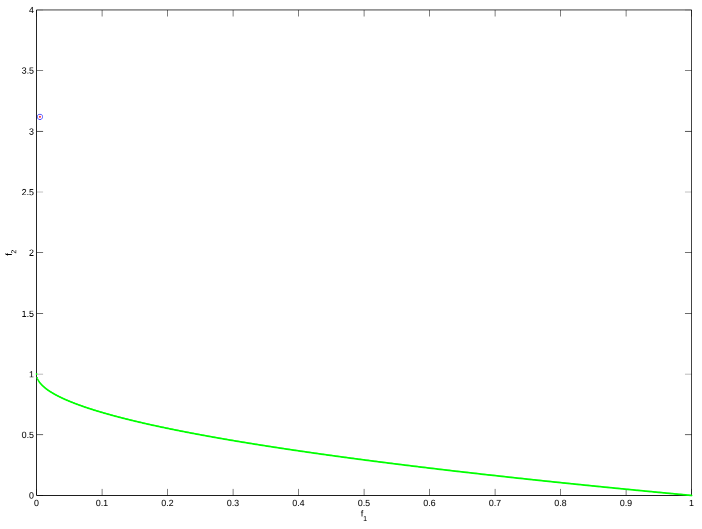


Figure 7.29: Problem ZDT_4 (zoomed in)

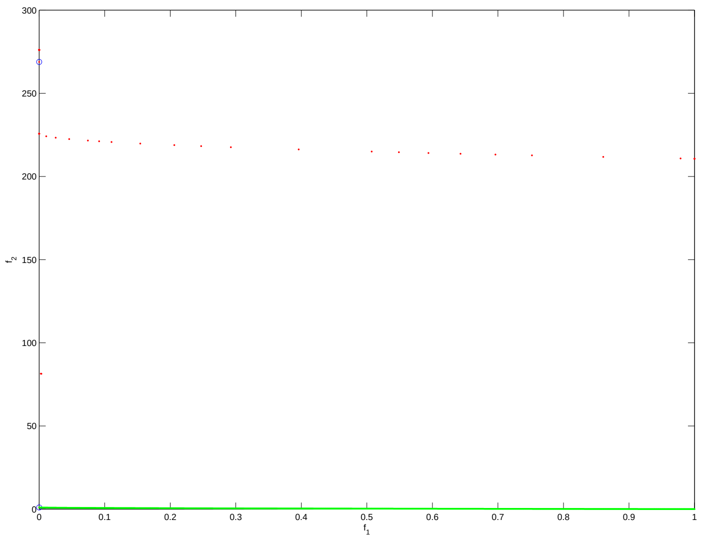


Figure 7.30: Problem ZDT_4_10

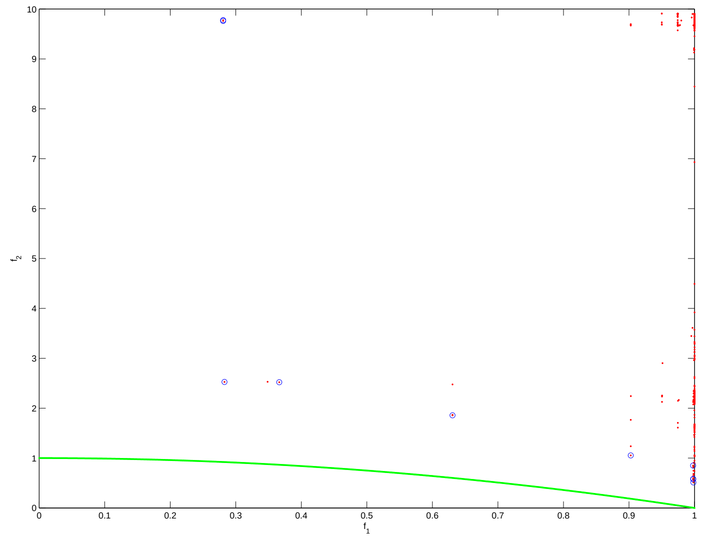


Figure 7.31: Problem ZDT_6

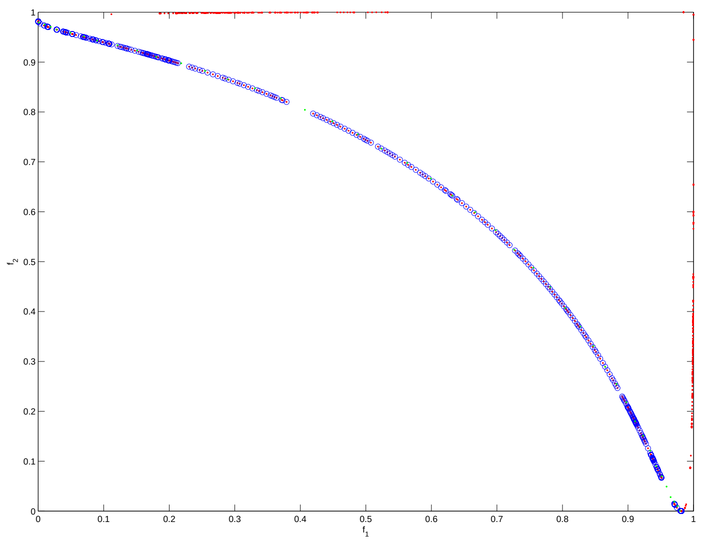


Figure 7.32: Problem Fonseca2

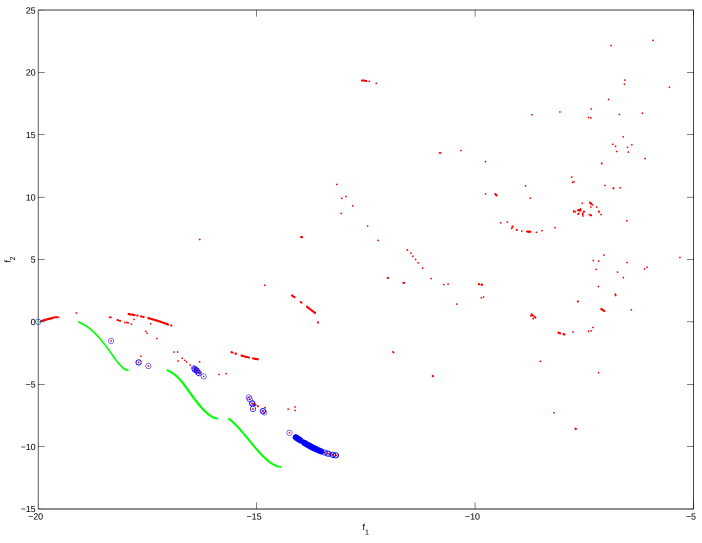


Figure 7.33: Problem Kursawe1

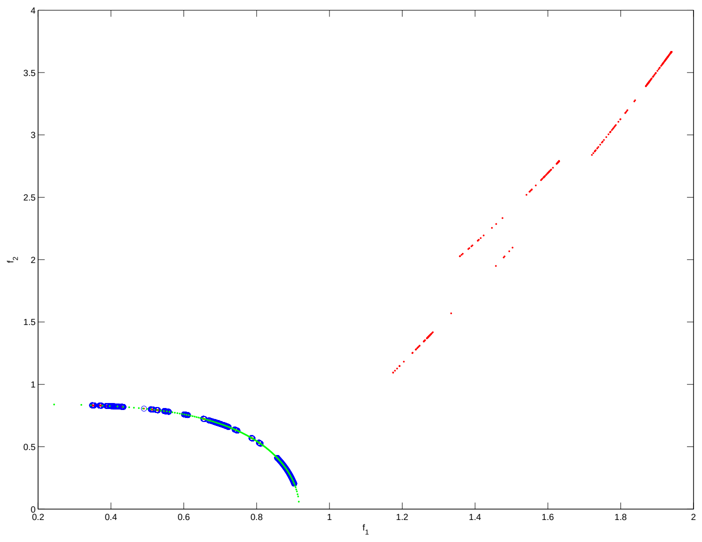


Figure 7.34: Problem Lis

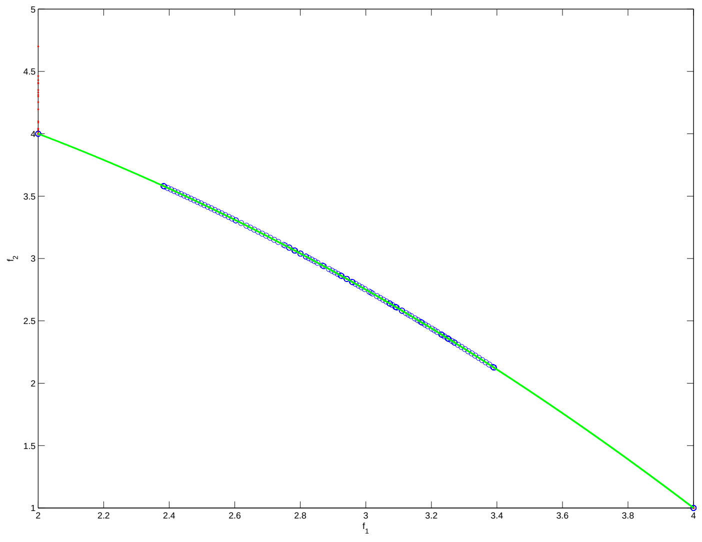


Figure 7.35: Problem Murata

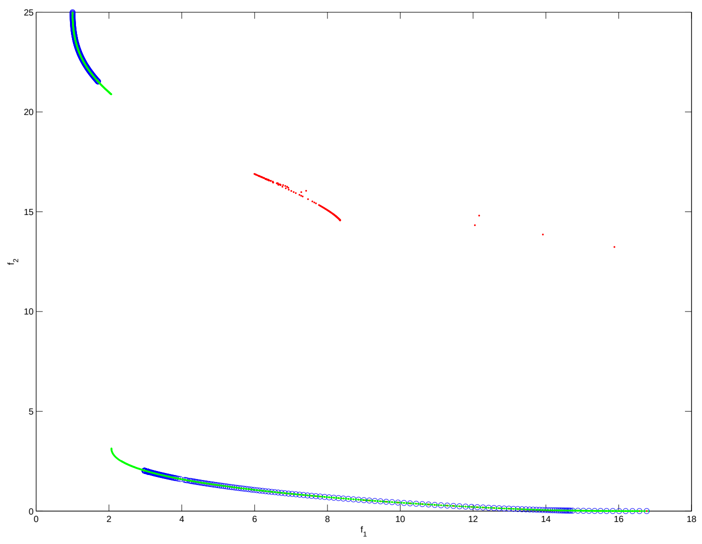


Figure 7.36: Problem Poloni

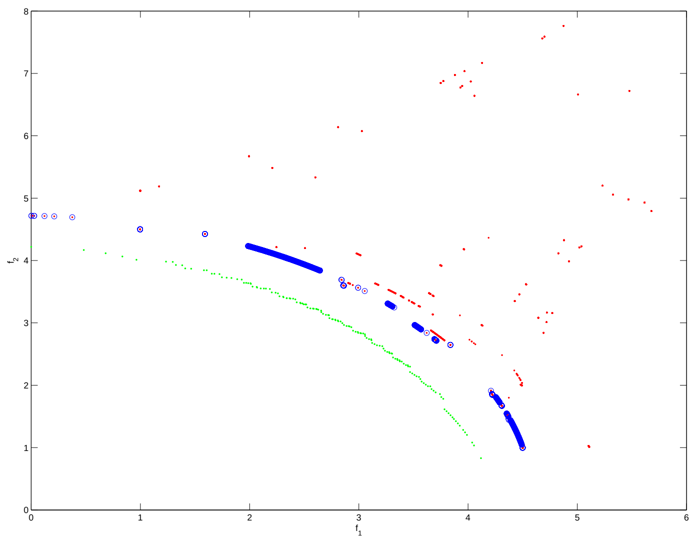


Figure 7.37: Problem Quagliarella

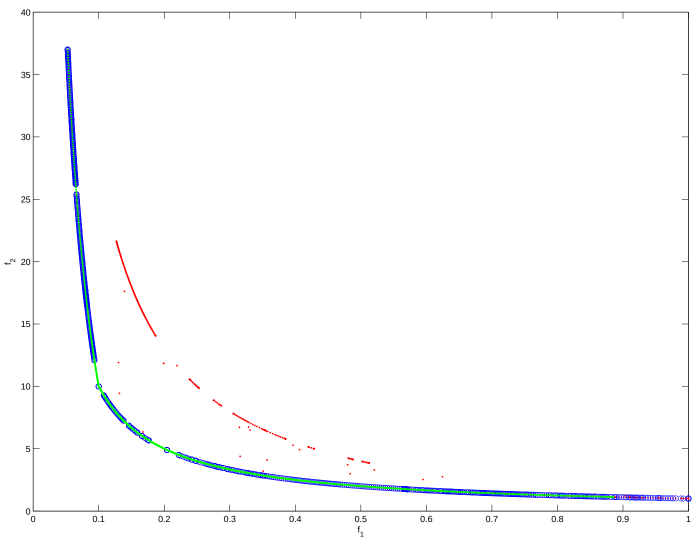


Figure 7.38: Problem Rendon

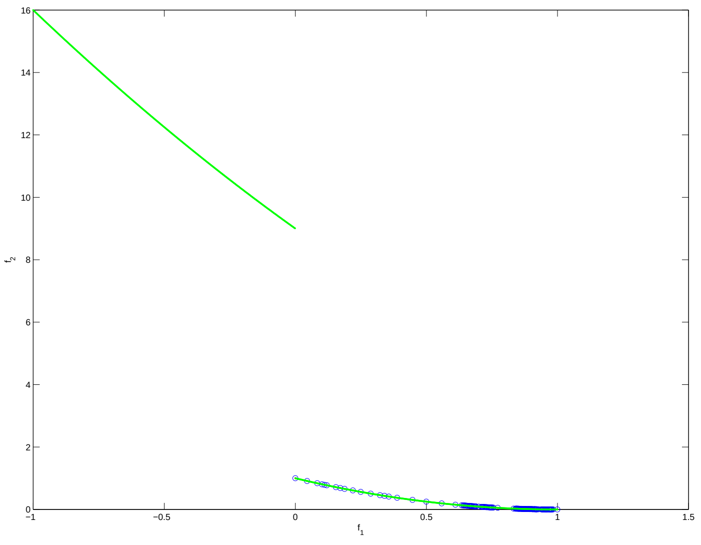


Figure 7.39: Problem Schaffer2

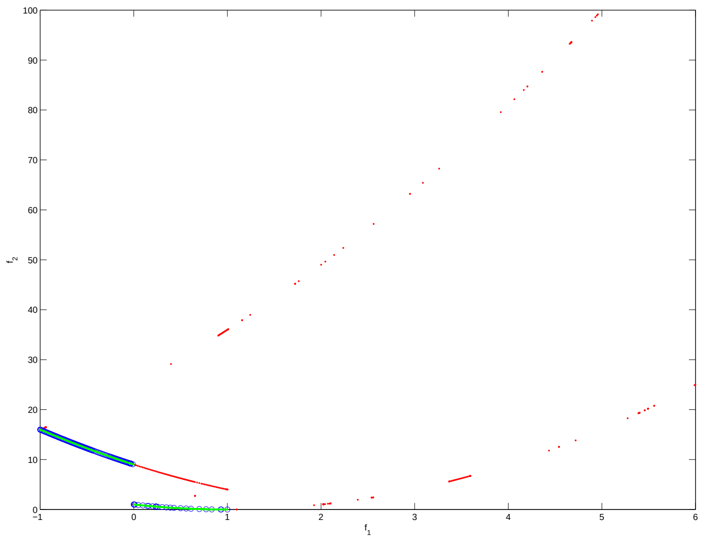


Figure 7.40: Problem Schaffer2_30

Table 7.7 provides a detailed insight into the operations of the computer code for this set of test problems. As expected, the performance of the algorithm deteriorates for this difficult class of instances. Nevertheless, we do obtain satisfying results for the test cases stemming from the Van Veldhuizen test suite as can be seen in Figures 7.32, 7.34, 7.35, 7.36, 7.38, 7.39 and 7.40. At this point we remark that for the instance `Schaffer2` we can compute an excellent approximation for the efficient set (cf. Figure 7.40) if we tinker with the starting point and set the initial value $\zeta = 30$. Similarly, if we change the starting point for problem `ZDT_4` and opt for $\zeta = 10$, we result in an approximation of a local Pareto optimal front (cf. Figure 7.30). A further observation worth highlighting is that the solution method manages to determine efficient points in all disconnected parts of the efficient curves for all problems with disconnected sets of efficient points `F11_G1_H4`, `ZDT_3_30`, `ZDT_3_100` and particularly `Poloni`. Moreover, we deduce from problem `ZDT_3` that an increase in the dimension of the solution vector does not pose a problem for our solution method since the performance measures remain of the same order. Besides we note that we utilised the squared objective functions approach for problem `F11_G1_H3` to determine the local concave efficient curve and also for the instance `ZDT_6`. Furthermore, we remark that we were not able to verify the actual efficient curve provided for the problem `Quagliarella` by Coello [4]. Finally, we witnessed very large numbers of gradient evaluations per point once again for these test problems, which emphasises the potential for improvement that could be realised with an improved line search routine for the algorithm NLIPF.

### 7.3.4 Warm-start strategy

In order to appreciate the contribution of the warm-start strategy to the NL_EFFTREE algorithm we want to highlight the two major effects this technique has on the performance of our method.

| Problem | time (sec) | points | $\varepsilon$-Pareto points (%) | SOLE / pt | Jacobian / pt | Function eval. / pt |
|---|---|---|---|---|---|---|
| F13_G1_H1 | 85.16 | 301 | 33.89 | 31.72 | 2482.04 | 31.67 |
| F14_G1_H1 | 7.61 | 362 | 83.98 | 6.06 | 150.22 | 5.44 |
| F11_G2_H1_05 | 290.48 | 1226 | 31.81 | 31.42 | 2128.39 | 30.45 |
| F11_G3_H1 | 10.16 | 708 | 7.06 | 5.03 | 101.20 | 4.52 |
| F11_G1_H3 | 81.09 | 3268 | 87.39 | 5.57 | 186.38 | 4.22 |
| F11_G1_H4 | 95.59 | 867 | 49.94 | 13.95 | 969.78 | 12.40 |
| ZDT_3_30 | 1384.92 | 2157 | 24.62 | 7.09 | 378.13 | 5.92 |
| ZDT_3_100 | 6249.70 | 2640 | 40.87 | 6.65 | 334.47 | 5.31 |
| ZDT_4 | 712.45 | 497 | 0.40 | 32.36 | 2194.16 | 32.35 |
| ZDT_6 | 999.44 | 594 | 1.68 | 46.14 | 2838.73 | 46.05 |
| Fonseca2 | 9.16 | 1223 | 43.42 | 2.16 | 26.06 | 1.46 |
| Kursawe1 | 160.58 | 2826 | 18.44 | 7.62 | 287.34 | 5.41 |
| Lis | 174.92 | 1784 | 69.45 | 15.49 | 900.04 | 14.55 |
| Murata | 9.06 | 508 | 99.61 | 6.50 | 120.24 | 4.73 |
| Poloni | 35.81 | 609 | 71.76 | 11.37 | 475.24 | 11.02 |
| Quagliarella | 259.77 | 3008 | 33.01 | 3.21 | 44.24 | 2.23 |
| Rendon | 29.56 | 1421 | 77.76 | 6.11 | 152.26 | 5.33 |
| Schaffer2 | 4.66 | 150 | 98.67 | 7.57 | 206.55 | 7.55 |

Table 7.7: Results for nonconvex test problems. Explanation of the columns: 1 - name of the test problem, 2 - running time, 3 - number of solution points, 4 - percentage of $\varepsilon$-Pareto optimal points of the solution points, 5 - number of systems of linear equations solved per solution point, 6 - number of Jacobian vector evaluations per solution point, 7 - number of function evaluations per solution point.

Firstly, the warm-start strategy automatically determines the different weighting parameters that lead to different solutions points. Without this mechanism we would have to devise an alternative approach to the difficult problem of choosing suitable weighting parameters.
Secondly, and more crucially, the warm-start strategy significantly increases the computational efficiency since it exploits iterates that are closer to the optimal solutions and less infeasible with regards to the side constraints than the initial starting point. To illustrate this effect with an example we analyse the test problem F11_G1_H1 in more details.
If we employ the warm-start strategy in the NL_EFFTREE algorithm to determine the solution set for problem F11_G1_H1 we have to solve 1777 systems of linear equations to compute 253 points (7.02 SOLE/pt). In particular, this comprises of 414 systems for the warm-start strategy, 193 systems for cold starts and 1170 systems as part of the NLIPF algorithm. In comparison, if we compute the solutions for the 253 different weighting parameters obtained from the warm-start strategy individually from the initial starting point we have to solve 5637 systems of linear equations as part of the NLIPF algorithm in total or 22.28 SOLE/pt. Hence, the warm-start strategy reduces the computational complexity by 68.5%.

# Chapter 8

# Conclusions And Future Work

In this thesis we devised an infeasible interior-point path-following algorithm employing a warm-start strategy to solve nonlinear multiobjective optimisation problems.

During the course of the development of this method we relied on several different concepts of mathematics. Initially the theoretical foundations of multiobjective optimisation were exploited to derive a strategy to determine the set of efficient solutions via scalarising functionals and the terminology of proper efficiency. Crucial in this aspect is the fact that the set of properly efficient points is dense in the set of efficient solutions under certain prerequisites. Thus, we were equipped with the tools to compute an approximation to the Pareto-optimal set by considering a sufficiently large number of scalar ersatz problems which can be uniquely identified by their corresponding weighting parameter.

Utilising the renowned necessary optimality conditions introduced by Karush, Kuhn and Tucker and applying a modified Newton method we established an interior-point path-following algorithm which would lead to a solution to a scalar ersatz problem for one particular weighting parameter. Moreover, we employed an infeasible variant of this technique which allows for an uncomplicated choice for the starting point and the iterates of our method. Furthermore, we presented comprehensive analysis concerning the convergence of our algorithm to an optimal solution. Based on our examination we had to specify the objective function in further details and demand additional characteristics from particular elements of the solution method.

In addition, we devised an adaptation of the elegant warm-start strategy which generates new iterates for different instances of the scalar ersatz problems via a perturbation mechanism. As a consequence of this utilisation of known information we were in a position to generate broad approximations of the efficient set at early stages of the algorithm and contribute to a reduction of the computational complexity of our approach.

Based on these theoretical foundations we developed a numerical implementation of our solution method using the software platform MATLAB. In order to investigate and evaluate the performance of our computer code we formulated a family of test problems based on the test suites by Deb, Zitzler et al. and Van Veldhuizen. We obtained impressive results from our computer code for the classes of convex and concave optimisation problems. Finally we observed the theoretical limitations of our approach by considering (highly) nonconvex test instances.

With regards to potential future extensions of our research findings these can be divided in theoretical advancements and improvements of the numerical implementation. Concerning the first domain one very interesting topic for example would be the incorporation of more nonlinear objective functions to the multiobjective optimisation problem and hence, enabling us to enhance our current model to higher dimensions. In particular the warm-start strategy would need extensive modifications since the weighting parameter $\omega$ would no longer be a scalar but a higher dimensional vector.

With regards to our computer code the performance analysis revealed that several areas of implementation of the algorithm show room for improvement. Firstly, a more effective routine handling the step length computation for the Newton system would be desirable. This requires a more sophisticated approach which we could not prioritise in this thesis due to time limitations. Secondly, we observed that the vast majority of the running time is spent in the routines generating the first and second order derivatives. Again, more advanced means of computing this information should be reachable but obviously were not the main focus of our research. Lastly, a brief examination of our abort criteria indicated that a reduction of the maximal number of loops allowed from 50 to 30 hardly affected the number of solution points attained, however drastically improved the computational time of our algorithm. Hence, a more thorough analysis of the parameters of our computer code based on a wider set of test problems and subsequent adjustments would be a potential next step in the development of our solution method.

# Appendix A
# Results From Analysis And Linear Algebra

In this section key results from analysis and linear algebra are recapitulated.

**Definition A.1 (Positive definite and positive semi-definite matrices)** *Let $A \in \mathbb{R}^{n\times n}$ be a square matrix. Then $A$ is called* ***positive semi-definite*** *if*

$$x^T Ax \geqslant 0 \quad \text{for all} \quad x \in \mathbb{R}^n.$$

*Furthermore, $A$ is denoted as* ***positive definite*** *if*

$$x^T Ax > 0 \quad x \in \mathbb{R}^n, x \neq 0.$$

**Theorem A.1** *Let $\Omega \subset \mathbb{R}^n$ be an open and convex set and $f$ be a twice differentiable function $f : \Omega \to \mathbb{R}$. Then*

- *$\nabla^2 f(x)$ is positive semi-definite for all $x \in \Omega$ if and only if $f$ is convex on $\Omega$*
- *if $\nabla^2 f(x)$ is positive definite for all $x \in \Omega$, $f$ is strictly convex on $\Omega$.*

*Proof.* Consult [20, pages 190–191]. □

**Theorem A.2** *Let $A \in \mathbb{R}^{n\times n}$ be a positive definite matrix and $B \in \mathbb{R}^{n\times n}$ be a positive semi-definite one. Then $A + B$ is a positive definite matrix.*

*Proof.* Stated in [30, page 151]. □

**Theorem A.3** *A positive definite matrix $A \in \mathbb{R}^{n\times n}$ is nonsingular. In other words, for every positive definite matrix $A$ the inverse matrix $A^{-1}$ exists.*

*Proof.* Stated in [30, page 152]. □

**Theorem A.4** *A square matrix $A \in \mathbb{R}^{n\times n}$ is nonsingular, i.e. the inverse $A^{-1} \in \mathbb{R}^{n\times n}$ exists if and only if the determinant of $A$ is non-zero. That is,*

$$A \text{ is nonsingular } \iff \det A \neq 0.$$

*Proof.* See [32, page 465]. □

**Theorem A.5** *Let $A, B \in \mathbb{R}^{n\times n}$ be square matrices. Then the following statement holds*

$$\det(AB) = \det(A)\det(B).$$

*Proof.* Compare [32, page 467]. □

**Corollary A.1** *Let $A, B \in \mathbb{R}^{n\times n}$ be square matrices. Then if $AB$ is nonsingular, also $A$ and $B$ are nonsingular.*

**Theorem A.6** *Let $A, B \in \mathbb{R}^{n\times n}$ be nonsingular matrices. Then the following equality is true*

$$(AB)^{-1} = B^{-1}A^{-1}.$$

**Theorem A.7** *Let $A \in \mathbb{R}^{m\times n}$ be a matrix with full row rank and $B \in \mathbb{R}^{n\times n}$ nonsingular. Then $ABA^T$ is also nonsingular and thus, invertible.*

*Proof.* Compare [30, pages 58 – 59]. □

# Appendix B
# List of test problems

**Problem name:** F11_G1_H1
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 + x_1 \\ g(x_2) &:= 1 + x_2 \\ h(f_1, g) &:= 1/f_1 \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F12_G1_H1_10
**Formulation:**

$$\begin{aligned} f_1(x_1, \ldots, x_{10}) &:= 1 + \textstyle\sum_{i=1}^{10} x_i \\ g(x_{11}) &:= 1 + x_{11} \\ h(f_1, g) &:= 1/f_1 \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F13_G1_H1
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 - \exp(-4x_1)\sin^4(5\pi x_1) \\ g(x_2) &:= 1 + x_2 \\ h(f_1, g) &:= 1/f_1 \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F14_G1_H1

**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 2.0 - \exp(-((x_1 - 0.2)/0.004)^2) - 0.8\exp(-((x_1 - 0.6)/0.4)^2) \\ g(x_2) &:= 1 + x_2 \\ h(f_1, g) &:= 1/f_1 \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F11_G2_H1_2
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 + x_1 \\ g(x_2) &:= 1 + x_2^2 \\ h(f_1, g) &:= 1/f_1 \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F11_G2_H1_05
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 + x_1 \\ g(x_2) &:= 1 + x_2^{0.5} \\ h(f_1, g) &:= 1/f_1 \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F11_G3_H1
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 + x_1 \\ g(x_2) &:= 1 + 20 + (x_2 - 30)^2 - 10\cos(2\pi(x_2 - 30)) \\ h(f_1, g) &:= 1/f_1 \end{aligned}$$

**Specifications:**

$$\begin{aligned} x_2 &\in [0, 60] \\ \zeta &= 600 \end{aligned}$$

**Problem name:** F11_G1_H1b_05

**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 + x_1 \\ g(x_2) &:= 1 + x_2 \\ h(f_1, g) &:= 1 - (f_1/11g)^{0.5} \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F11_G1_H1b_1
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 + x_1 \\ g(x_2) &:= 1 + x_2 \\ h(f_1, g) &:= 1 - (f_1/11g)^{1} \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F11_G1_H2
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 1 + x_1 \\ g(x_2) &:= 1 + x_2 \\ h(f_1, g) &:= 1 - (f_1/11g)^{2} \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** F11_G1_H3
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 4x_1 \\ g(x_2) &:= \begin{cases} 4 - 3\exp(-((x_2 - 0.2)/0.02)^2) & \text{if } 0 \leqslant x_2 \leqslant 0.4 \\ 4 - 2\exp(-((x_2 - 0.7)/0.2)^2) & \text{if } 0.4 < x_2 \leqslant 1 \end{cases} \\ h(f_1, g) &:= \begin{cases} 1 - (f_1/g)^{\alpha} & \text{if } f_1 \leqslant g \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

**Specifications:**

$$\begin{aligned} \alpha &= 0.25 + 3.75(g(x_2) - 1) \\ \zeta &= 10 \end{aligned}$$

**Problem name:** F11_G1_H4

**Formulation:**

$$\begin{aligned} f_1(x_1) &:= x_1 \\ g(x_2) &:= 1 + 10x_2 \\ h(f_1, g) &:= 1 - (f_1/g)^2 - (f_1/g)\sin(8\pi f_1) \end{aligned}$$

**Specifications:**

$$\zeta = 10$$

**Problem name:** ZDT_1
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= x_1 \\ g(x_2, \ldots, x_m) &:= 1 + 9 \cdot \textstyle\sum_{i=2}^{m} x_i/(m-1) \\ h(f_1, g) &:= 1 - \sqrt{f_1/g} \end{aligned}$$

**Specifications:**

$$\begin{aligned} m &= 30 \text{ and } 100 \\ \zeta &= 10 \end{aligned}$$

**Problem name:** ZDT_2
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= x_1 \\ g(x_2, \ldots, x_m) &:= 1 + 9 \cdot \textstyle\sum_{i=2}^{m} x_i/(m-1) \\ h(f_1, g) &:= 1 - (f_1/g)^2 \end{aligned}$$

**Specifications:**

$$\begin{aligned} m &= 30 \text{ and } 100 \\ \zeta &= 10 \end{aligned}$$

**Problem name:** ZDT_3
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= x_1 \\ g(x_2, \ldots, x_m) &:= 1 + 9 \cdot \textstyle\sum_{i=2}^{m} x_i/(m-1) \\ h(f_1, g) &:= 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1) \end{aligned}$$

**Specifications:**

$$\begin{aligned} m &= 30 \text{ and } 100 \\ \zeta &= 10 \end{aligned}$$

**Problem name:** ZDT_4

**Formulation:**

$$\begin{aligned}
f_1(x_1) &:= x_1\\
g(x_2,\dots,x_m) &:= 1+10(m-1)+\textstyle\sum_{i=2}^{m}((x_i-5)^2-10\cos(4\pi(x_i-5)))\\
h(f_1,g) &:= 1-\sqrt{f_1/g}
\end{aligned}$$

**Specifications:**

$$\begin{aligned}
m &= 10\\
x_2,\dots,x_m &\in [0,10]\\
\zeta &= 100
\end{aligned}$$

**Problem name:** ZDT_6
**Formulation:**

$$\begin{aligned}
f_1(x_1) &:= 1-\exp(-4x_1)\sin^6(6\pi x_1)\\
g(x_2,\dots,x_m) &:= 1+9\cdot((\textstyle\sum_{i=2}^{m}x_i)/(m-1))^{0.25}\\
h(f_1,g) &:= 1-(f_1/g)^2
\end{aligned}$$

**Specifications:**

$$\begin{aligned}
m &= 10\\
\zeta &= 10
\end{aligned}$$

**Problem name:** Binh1
**Formulation:**

$$\begin{aligned}
f_1(x_1,x_2) &:= (x_1-5)^2+(x_2-5)^2\\
f_2(x_1,x_2) &:= (x_1-10)^2+(x_2-10)^2
\end{aligned}$$

**Specifications:**

$$\begin{aligned}
x_1,x_2 &\in [0,15]\\
\zeta &= 150
\end{aligned}$$

**Problem name:** Fonseca
**Formulation:**

$$\begin{aligned}
f_1(x_1,x_2) &:= 1-\exp(-(x_1-101)^2-(x_2-99)^2)\\
f_2(x_1,x_2) &:= 1-\exp(-(x_1-99)^2-(x_2-101)^2)
\end{aligned}$$

**Specifications:**

$$\begin{aligned}
x_1,x_2 &\in [0,200]\\
\zeta &= 2000
\end{aligned}$$

**Problem name:** Fonseca2

**Formulation:**

$$\begin{aligned} f_1(x_1,\ldots,x_3) :=&\ 1-\exp(-\textstyle\sum_{i=1}^{3}(x_i-4-1/\sqrt{3})^2) \\ f_2(x_1,\ldots,x_3) :=&\ 1-\exp(-\textstyle\sum_{i=1}^{3}(x_i-4+1/\sqrt{3})^2) \end{aligned}$$

**Specifications:**

$$\begin{aligned} x_1,\ldots,x_3 &\in [0,8] \\ \zeta &= 80 \end{aligned}$$

**Problem name:** Kursawe1
**Formulation:**

$$\begin{aligned} f_1(x_1,\ldots,x_3) :=&\ \textstyle\sum_{i=1}^{2} -10\exp(-0.2\sqrt{(x_i-5)^2+(x_{i+1}-5)^2}) \\ f_2(x_1,\ldots,x_3) :=&\ \textstyle\sum_{i=1}^{3}(|x_i-5|^{0.8}+5\sin(x_i-5)^3) \end{aligned}$$

**Specifications:**

$$\begin{aligned} x_1,\ldots,x_3 &\in [0,10] \\ \zeta &= 100 \end{aligned}$$

**Problem name:** Laumanns
**Formulation:**

$$\begin{aligned} f_1(x_1,x_2) :=&\ (x_1-50)^2+(x_2-50)^2 \\ f_2(x_1,x_2) :=&\ (x_1-48)^2+(x_2-50)^2 \end{aligned}$$

**Specifications:**

$$\begin{aligned} x_1,x_2 &\in [0,100] \\ \zeta &= 1000 \end{aligned}$$

**Problem name:** Lis
**Formulation:**

$$\begin{aligned} f_1(x_1,x_2) :=&\ \sqrt[8]{(x_1-5)^2+(x_2-5)^2} \\ f_2(x_1,x_2) :=&\ \sqrt[4]{(x_1-5.5)^2+(x_2-5.5)^2} \end{aligned}$$

**Specifications:**

$$\begin{aligned} x_1,x_2 &\in [0,15] \\ \zeta &= 150 \end{aligned}$$

**Problem name:** Murata
**Formulation:**

$$\begin{aligned} f_1(x_1) &:= 2\sqrt{x_1+1} \\ f_2(x_1,x_2) &:= (x_1+1)(-x_2)+5 \end{aligned}$$

**Specifications:**

$$\begin{array}{lll} x_1 & \in & [0,3] \\ x_2 & \in & [0,1] \\ \zeta & = & 30 \end{array}$$

**Problem name:** Poloni
**Formulation:**

$$\begin{array}{ll} f_1(x_1,x_2) := & 1 + (A_1 - B_1)^2 + (A_2 - B_2)^2 \\ f_2(x_1,x_2) := & (x_1 - \pi + 3)^2 + (x_2 - \pi + 1)^2 \end{array}$$

**Specifications:**

$$\begin{array}{lll} A_1 & = & 0.5 \sin 1 - 2 \cos 1 + \sin 2 - 1.5 \cos 2 \\ A_2 & = & 1.5 \sin 1 - \cos 1 + 2 \sin 2 - 0.5 \cos 2 \\ B_1 & = & 0.5 \sin(x_1 - \pi) - 2 \cos(x_1 - \pi) + \sin(x_2 - \pi) - 1.5 \cos(x_2 - \pi) \\ B_2 & = & 1.5 \sin(x_1 - \pi) - \cos(x_1 - \pi) + 2 \sin(x_2 - \pi) - 0.5 \cos(x_2 - \pi) \\ x_1, x_2 & \in & [0, 2\pi] \\ \zeta & = & 20\pi \end{array}$$

**Problem name:** Quagliarella
**Formulation:**

$$\begin{array}{ll} f_1(x_1,\ldots,x_{16}) := & \sqrt{(\sum_{i=1}^{16}((x_i - 5.12)^2 - 10 \cos(2\pi(x_i - 5.12)) + 10))/16} \\ f_2(x_1,\ldots,x_{16}) := & \sqrt{(\sum_{i=1}^{16}((x_i - 6.62)^2 - 10 \cos(2\pi(x_i - 6.62)) + 10))/16} \end{array}$$

**Specifications:**

$$\begin{array}{lll} x_1,\ldots,x_{16} & \in & [0, 10.24] \\ \zeta & = & 102.4 \end{array}$$

**Problem name:** Rendon
**Formulation:**

$$\begin{array}{ll} f_1(x_1,x_2) := & 1/((x_1 - 3)^2 + (x_2 - 3)^2 + 1) \\ f_2(x_1,x_2) := & (x_1 - 3)^2 + 3(x_2 - 3)^2 + 1 \end{array}$$

**Specifications:**

$$\begin{array}{lll} x_1, x_2 & \in & [0,6] \\ \zeta & = & 60 \end{array}$$

**Problem name:** Rendon2

**Formulation:**

$$\begin{aligned} f_1(x_1,x_2) &:= x_1 + x_2 - 5 \\ f_2(x_1,x_2) &:= (x_1-3)^2 + 2(x_2-3) - 1 \end{aligned}$$

**Specifications:**

$$\begin{aligned} x_1, x_2 &\in [0,6] \\ \zeta &= 60 \end{aligned}$$

**Problem name:** Schaffer
**Formulation:**

$$\begin{aligned} f_1(x) &:= (x-10^5)^2 \\ f_2(x) &:= (x-10^5-2)^2 \end{aligned}$$

**Specifications:**

$$\begin{aligned} x &\in [0, 2\cdot 10^5] \\ \zeta &= 2\cdot 10^6 \end{aligned}$$

**Problem name:** Schaffer2
**Formulation:**

$$\begin{aligned} f_1(x) &:= \begin{cases} 5-x & \text{if } x \leqslant 6 \\ x-7 & \text{if } 6 < x \leqslant 8 \\ 9-x & \text{if } 8 < x \leqslant 9 \\ x-9 & \text{if } 9 < x \end{cases} \\ f_2(x) &:= (x-10)^2 \end{aligned}$$

**Specifications:**

$$\begin{aligned} x &\in [0,15] \\ \zeta &= 150 \end{aligned}$$

**Problem name:** Kita
**Formulation:**

$$\begin{aligned} f_1(x_1,x_2) &:= x_1^2 - x_2 \\ f_2(x_1,x_2) &:= -0.5x_1 - x_2 - 1 \end{aligned}$$

**Specifications:**

$$\begin{aligned} x_1/6 + x_2 &\leqslant 13/2 \\ x_1/2 + x_2 &\leqslant 15/2 \\ 5x_1 + x_2 &\leqslant 30 \\ \zeta &= 65 \end{aligned}$$

# List of References

[1] P. R. Adby and M. A. H. Dempster. *Introduction to Optimization Methods*. Chapman and Hall, London, 1974.

[2] Uttarayan Bagchi. Simultaneous minimization of mean and variation of flow time and waiting time in single machine systems. *Oper. Res.*, 37(1):118–125, 1989.

[3] Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear Programming*. John Wiley & Sons, Inc., New York, second edition, 1993. Theory and Algorithms.

[4] Coello. Evolutionary Algorithms for Solving Multi-Objective Problems - Appendix B. http://www.cs.cinvestav.mx/ emoobook/apendix-b/apendix-b.html.

[5] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-objective Problems*. Genetic and Evolutionary Computation Series. Springer, New York, second edition, 2007. With a foreword by David E. Goldberg.

[6] Kalyanmoy Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230, 1999.

[7] Francis Y. Edgeworth. *Mathematical Psychics*. P. Keagan, London, 1881.

[8] Matthias Ehrgott. *Multicriteria Optimization*. Springer-Verlag, Berlin, second edition, 2005.

[9] Gabriele Eichfelder. *Parameter Controlled Solutions of Nonlinear Multi-Objective Optimization Problems (in German)*. PhD thesis, Department of Mathematics, University of Erlangen-Nürnberg.

[10] Gerald W. Evans. An overview of techniques for solving multiobjective mathematical programs. *Management Sci.*, 30(11):1268–1282, 1984.

[11] Peter C. Fishburn. *Mathematics of Decision Theory (Methods and models in the social sciences)*. Mouton, 1972.

[12] Jörg Fliege. Olaf - a general modeling system to evaluate and optimize the location of an air polluting facility. *ORSpektrum*, 23(1):117–136, 2001.

[13] Jörg Fliege and Andree Heseler. *Constructing Approximations to the Efficient Set of Convex Quadratic Multiobjective Problems*. Ergebnisberichte Angewandte Mathematik, No. 211, Fachbereich Mathematik, Universität Dortmund, 2002.

[14] Jörg Fliege and Luis N. Vicente. Multicriteria approach to bilevel optimization. *Journal of Optimization Theory and Applications*, 131(2):209–225, 2006.

[15] Yan Fu and Urmila M. Diwekar. An efficient sampling approach to multiobjective optimization. *Annals of Operations Research*, 132:109–134, 2004.

[16] Alfred Göpfert and Reinhard Nehse. *Vektoroptimierung. Theorie, Verfahren und Anwendungen*, volume 74 of *Mathematisch-Naturwissenschaftliche Bibliothek [Mathematical-Scientific Library]*. BSB B. G. Teubner Verlagsgesellschaft, Leipzig, 1990.

[17] M. Gravel, I. M. Martel, R. Madeau, W. Price, and R. Tremblay. A multicriterion view of optimal ressource allocation in job-shop production. *European Journal of Operation Research*, 61:230–244, 1992.

[18] Christoph Heermann. *Innere-Punkt-Methoden in der mulitkriteriellen Kraftwerksoptimierung*. Diplomarbeit, Fachbereich Mathematik, Universität Dortmund, 2003.

[19] Claus Hillermeier. *Nonlinear Multiobjective Optimization. A Generalized Homotopy Approach*, volume 135 of *International Series of Numerical Mathematics*. Birkhäuser Verlag, Basel, 2001.

[20] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex Analysis and Minimization Algorithms. I*, volume 305 of *Grundlehren der Mathematischen Wissenschaften [A Series of Comprehensive Studies in Mathematics]*. Springer-Verlag, Berlin, 1993.

[21] Simon Huband, Philip Hingston, Luigi Barone, and R. Lyndon While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006.

[22] Angelika Hutterer and Johannes Jahn. Optimization of the location of antennas for treatment planning in hyperthermia. Technical Report Preprint 265, Institut für Angewandte Mathematik, Universität Erlangen-Nürnberg, Martensstrasse 3, D-91058 Erlangen, Germany, 2000.

[23] Elizabeth John and E. Alper Yıldırım. Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimension. *Computational Optimization and Applications. An International Journal*, 41(2):151–183, 2008.

[24] Arno Jüschke, Johannes Jahn, and Andreas Kirsch. A bicriterial optimization problem of antenna design. *Comput. Optim. Appl.*, 7(3):261–276, 1997.

[25] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[26] Leonid G. Khachiyan. A polynomial algorithm in linear programming. *Dokl. Akad. Nauk SSSR*, 244(5), 1979. English translation: Soviet Math. Dokl. 20 (1979), No. 1, 191–194.

[27] Hellmuth Kneser. Eine direkte Ableitung des Zornschen Lemmas aus dem Auswahlaxiom. *Mathematische Zeitschrift*, 53:110–113, 1950.

[28] Harold W. Kuhn and Albert W. Tucker. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950*, pages 481–492, Berkeley and Los Angeles, 1951. University of California Press.

[29] David G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley Publishing Company, Reading, Massachusetts, second edition, 1984.

[30] Helmut Lütkepohl. *Handbook of Matrices.* John Wiley & Sons, Inc., Chichester, 1996.

[31] John McCleary. *A First Course in Topology*, volume 31 of *Student Mathematical Library.* American Mathematical Society, Providence, Rhode Island, 2006. Continuity and Dimension.

[32] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pennsylvania, 2000.

[33] Daniel Molz, Christoph Heermann, and Jörg Fliege. An adaptive primal-dual warm-start technique for quadratic multiobjective optimization. Technical Report Preprint 2006/34, School of Mathematics, The University of Birmingham, Edgbaston, Birmingham B15 2TT, UK, 2006.

[34] Renato D. C. Monteiro and Fangjun Zhou. On the existence and convergence of the central path for convex programming and some duality results. *Comput. Optim. Appl.*, 10(1):51–77, 1998.

[35] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization.* Springer Series in Operations Research. Springer-Verlag, New York, 1999.

[36] Vilfredo Pareto. *Manuale di Economia Politica.* Societa Editrice Libraria, Milano, 1906. Translated by Ann S. Schwier as Manual Of Political Economy, The MacMillian Press LTD, New York, 1971.

[37] D. Prabuddha, J. B. Gosh, and C.E. Wells. On the minimization of completion time variance with a bicriteria extension. *Operations Research*, 40:1148–1155, 1992.

[38] R. Tyrrell Rockafellar. *Convex Analysis.* Princeton Mathematical Series, No. 28. Princeton University Press, Princeton, New Jersey, 1970.

[39] Yoshikazu Sawaragi, Hirotaka Nakayama, and Tetsuzo Tanino. *Theory of multiobjective optimization*, volume 176 of *Mathematics in Science and Engineering*. Academic Press Inc., Orlando, Florida, 1985.

[40] Songqinq Shan and G. Gary Wang. An efficient pareto set identification approach for multiobjective optimization on black-box functions. *Journal of Mechanical Design*, 127:866–874, 2005.

[41] Wolfram Stadler. A survey of multicriteria optimization or the vector maximum problem. I. 1776–1960. *Journal of Optimization Theory and Applications*, 29(1):1–52, 1979.

[42] David A. Van Veldhuizen and Gary B. Lamont. Multiobjective evolutionary algorithm research: A history and analysis. Technical Report TR-98-03, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1998.

[43] David A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Wright-Patterson AFB, Ohio, 1999.

[44] Andreas Wächter and Lorenz T. Biegler. Line search filter methods for nonlinear programming: motivation and global convergence. *SIAM Journal on Optimization*, 16(1):1–31 (electronic), 2005.

[45] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1, Ser. A):25–57, 2006.

[46] Yan-jin Wang and Pu-sheng Fei. *A Primal-Infeasible Interior Point Algorithm For Linearly Constrained Convex Programming*. http://www.optimization-online.org, 2005.

[47] D. J. White. Epsilon-dominating solutions in mean-variance portfolio analysis. *European Journal of Operational Research*, 105:457–466, 1998.

[48] Stephen J. Wright. *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pennsylvania, 1997.

[49] E. Alper Yildirim and Stephen J. Wright. Warm-start strategies in interior-point methods for linear programming. *SIAM Journal on Optimization*, 12(3):782–810, 2002.

[50] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

[51] Max Zorn. A remark on method in transfinite algebra. *Bulletin of the American Mathematical Society*, 41(10):667–670, 1935.