Multicriteria Optimization and Decision Making

Principles, Algorithms and Case Studies

Michael Emmerich and André Deutz
LIACS Master Course: Autumn/Winter 2014/2015

Contents

1 Introduction **5**
1.1 Viewing mulicriteria optimization as a task in system design and analysis . . . 7
1.2 Formal Problem Definitions . . . 9
1.2.1 Other Problem Classes in Optimization . . . 10
1.2.2 Multiobjective Optimization . . . 11
1.3 Problem Difficulty in Optimization . . . 12
1.3.1 Problem Difficulty in Continuous Optimization . . . 12
1.3.2 Problem Difficulty in Combinatorial Optimization . . . 14
1.4 Pareto dominance and incomparability . . . 17
1.5 Formal Definition of Pareto Dominance . . . 19

I Foundations **21**

2 Orders and dominance **22**
2.1 Preorders . . . 22
2.2 Preorders . . . 23
2.3 Partial orders . . . 25
2.4 Linear orders and anti-chains . . . 26
2.5 Hasse diagrams . . . 26
2.6 Comparing ordered sets . . . 28
2.7 Cone orders . . . 29

3 Landscape Analysis **34**
3.1 Search Space vs. Objective Space . . . 34
3.2 Global Pareto Fronts and Efficient Sets . . . 36
3.3 Weak efficiency . . . 37
3.4 Characteristics of Pareto Sets . . . 38
3.5 Optimality conditions based on level sets . . . 39
3.6 Local Pareto Optimality . . . 41

3.7 Barrier Structures . . . 43
3.8 Shapes of Pareto Fronts . . . 46
3.9 Conclusions . . . 49

4 Optimality conditions for differentiable problems 50
4.1 Linear approximations . . . 50
4.2 Unconstrained Optimization . . . 50
4.3 Equality Constraints . . . 52
4.4 Inequality Constraints . . . 55
4.5 Multiple Objectives . . . 57

5 Scalarization Methods 59
5.1 Linear Aggregation . . . 60
5.2 Nonlinear Aggregation . . . 62
5.3 Multi-Attribute Utility Theory . . . 62
5.4 Distance to a Reference Point Methods . . . 67

6 Transforming Multicriteria into Constrained Single-Criterion Problems 71
6.1 Compromise Programming or ϵ-Constraint Methods . . . 71
6.2 Concluding remarks on single point methods . . . 73

II Algorithms for Pareto Optimization 76

7 Pareto Front Computing with Deterministic Methods 77
7.1 Computing the Maximal Set of Vectors . . . 77

8 Evolutionary Multicriterion Optimization 81
8.1 Single Objective Evolutionary Algorithms . . . 82
8.2 Evolutionary Multiobjective Optimization . . . 86
8.2.1 Algorithms Balancing Dominance and Diversity . . . 88
8.2.2 Indicator-based Algorithms . . . 89
8.2.3 Multiobjective Evolution Strategy . . . 92
8.3 Final Remarks . . . 93

9 Conclusion 94

Preface

Finding optimal solutions in large and constrained search spaces has been since long the core topic of operations research and engineering optimization. Such problems are typically challenging from an algorithmic point of view. Multicriteria optimization can be seen as a modern variant of it, that also takes into account that in real world problems we also have to deal with multicriteria decision making and the aspect of searching for an optimum should be combined with aspects of multicriteria decision analysis (MCDA), which is the science of making good choices based on the systematic analysis of alternatives:

Real world decision and optimization problems usually involve conflicting criteria. Think of chosing a means to travel from one country to another. It should be fast, cheap or convenient, but you probably cannot have it all. Or you would like to design an industrial process, that should be safe, environmental friendly and cost efficient. Ideal solutions, where all objectives are at their optimal level, are rather the exception than the rule. Rather we need to find good compromises, and avoid lose-lose situations.

These lecture nodes deal with Multiobjective Optimization and Decision Analysis (MODA). We define this field, based on some other scientific disciplines:

DEF *Multicriteria Decision Aiding (MCDA)* (or: Multiattribute Decision Analysis) is a scientific field that studies evaluation of a finite number of alternatives based on multiple criteria. It provides methods to compare, evaluate, and rank solutions.

DEF *Multicriteria Optimization (MCO)* (or: Multicriteria Design, Multicriteria Mathematical Programming) is a scientific field that studies search for optimal solutions given multiple criteria and constraints. Here, usually, the search space is very large and not all solutions can be inspected.

DEF *Multicriteria Decision Making (MCDM)* deals with MCDA and MCO or combinations of these.

We use here the title: **Multicriteria Optimization and Decision Analysis = MODA** as a synonym of MCDM in order to focus more on the algorithmically challenging optimization aspect.

In this course we will deal with algorithmic methods for solving (constrained) multi-objective optimization and decision making problems. The rich mathematical structure of such problems as well as their high relevance in various application fields led recently to a significant increase of research activities. In particular algorithms that make use of fast, parallel computing technologies are envisaged for tackling hard combinatorial and/or nonlinear application problems. In the course we will discuss the theoretical foundations of multi-objective optimization problems and their solution methods, including order and decision theory, analytical, interactive and meta-heuristic solution methods as well as state-of-the-art tools for their performance-assessment. Also an overview on decision aid tools and formal ways to reason about conflicts will be provided. All theoretical concepts will be accompanied by illustrative hand calculations and graphical visualizations during the course. In the second part of the course, the discussed approaches will be exemplified by the presentation of case studies from the literature, including various application domains of decision making, e.g. economy, engineering, medicine or social science.

This reader is covering the topic of Multicriteria Optimization and Decision Making. Our aim is to give a broad introduction to the field, rather than to specialize on certain types of algorithms and applications. Exact algorithms for solving optimization algorithms are discussed as well as selected techniques from the field of metaheuristic optimization, which received growing popularity in recent years. The lecture notes provides a detailed introduction into the foundations and a starting point into the methods and applications for this exciting field of interdisciplinary science. Besides orienting the reader about state-of-the-art techniques and terminology, references are given that invite the reader to further reading and point to specialized topics.

Chapter 1

Introduction

For several reasons multicriteria optimization and decision making is an exciting field of computer science and operations research. Part of its fascination stems from the fact that in MCO and MCDM different scientific fields are addressed. Firstly, to develop the general foundations and methods of the field one has to deal with structural sciences, such as algorithmics, relational logic, operations research, and numerical analysis:

- How can we state a decision/optimization problem in a formal way?
- What are the essential differences between single objective and multiobjective optimization?
- How can we rank solutions? What different types of orderings are used in decision theory and how are they related to each other?
- Given a decision model or optimization problem, which formal conditions need to be satisfied for solutions to be optimal?
- How can we construct algorithms that obtain optimal solutions, or approximations to them, in an efficient way?
- What is the geometrical structure of solution sets for problems with more than one optimal solution?

Whenever it comes to decision making in the real world, these decisions will be made by people responsible for it. In order to understand how people come to decisions and how the psychology of individuals (cognition, individual decision making) and organizations (group decision making) needs to be studied. Questions like the following may arise:

- What are our goals? What makes it difficult to state goals? How do people define goals? Can the process of identifying goals be supported?
- Which different strategies are used by people to come to decisions? How can satisfaction be measured? What strategies are promising in obtaining satisfactory decisions?
- What are the cognitive aspects in decision making? How can decision support systems be build in a way that takes care of cognitive capabilities and limits of humans?
- How do groups of people come to decisions? What are conflicts and how can they be avoided? How to deal with minority interests in a democratic decision process? Can these aspects be integrated into formal decision models?

Moreover, decisions are always related to some real world problem. Given an application field, we may find very specific answers to the following questions:

- What is the set of alternatives?
- By which means can we retrieve the values for the criteria (experiments, surveys, function evaluations)? Are there any particular problems with these measurements (dangers, costs), and how to deal with them? What are the uncertainties in these measurements?
- What are the problem-specific objectives and constraints?
- What are typical decision processes in the field, and what implications do they have for the design of decision support systems?
- Are there existing problem-specific procedures for decision support and optimization, and what about the acceptance and performance of these procedures in practice?

In summary, this list of questions gives some kind of bird's eye view of the field. However, in these lecture notes we will mainly focus on the structural aspects of multi-objective optimization and decision making. On the other hand, we also devote one chapter to human-centric aspects of decision making and one chapter to the problem of selecting, adapting, and evaluating MOO tools for application problems.

1.1 Viewing mulicriteria optimization as a task in system design and analysis

The discussion above can be seen as a rough sketch of questions that define the scope of multicriteria optimization and decision making. However, it needs to be clarified more precisely what is going to be the focus of these lecture notes. For this reason we want to approach the problem class from the point of view of system design and analysis. Here, with system analysis, we denote the interdisciplinary research field, that deals with the modeling, simulation, and synthesis of complex systems.

Beside experimentation with a physical system, often a system model is used. Nowadays, system models are typically implemented as computer programs that solve (differential) equation systems, simulate interacting automata, or stochastic models. We will also refer to them as *simulation models*. An example for a simulation model based on differential equations would be the simulation of the fluid flow around an airfoil based on the Navier Stokes equations. An example for a stochastic system model, could be the simulation of a system of elevators, based on some agent based stochastic model.

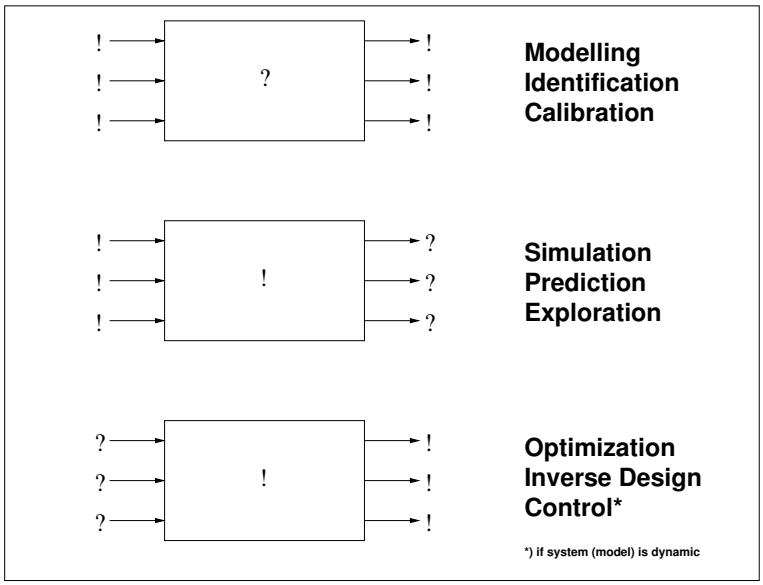

Figure 1.1: Different tasks in systems analysis.

In Figure 1.1 different tasks of systems analysis based on simulation models are displayed in a schematic way. *Modeling* means to identify the internal structure of the simulation model. This is done by looking at the relationship between known

inputs and outputs of the system. In many cases, the internal structure of the system is already known up to a certain granularity and only some parameters need to be identified. In this case we usually speak of *calibration* of the simulation model instead of modeling. In control theory, also the term *identification* is common.

Once a simulation-model of a system is given, we can simulate the system, i.e. predict the state of the output variables for different input vectors. Simulation can be used for predicting the output for not yet measured input vectors. Usually such model-based predictions are much cheaper than to do the experiment in the real world. Consider for example crash test simulations or the simulation of wind channels. In many cases, such as for future predictions, where time is the input variable, it is even impossible to do the experiments in the physical world. Often the purpose of simulation is also to learn more about the behavior of the systems. In this case systematic experimenting is often used to study effects of different input variables and combinations of them. The field of Design and Analysis of Computer Experiments (DACE) is devoted to such systematic explorations of a systems behavior.

Finally, we may want to optimize a system: In that case we basically specify what the output of the system should be. We also are given a simulation-model to do experiments with, or even the physical system itself. The relevant, open question is how to choose the input variables in order to achieve the desired output. In optimization we typically want to maximize (or minimize) the value of an output variable.

On the other hand, a very common situation in practice is the task of adjusting the value of an output variable in a way that it is as close as possible to a desired output value. In that case we speak about inverse design, or if the system is dynamically changing, it may be classified as a optimal control task. An example for an inverse design problem is given in airfoil design, where a specified pressure profile around an airfoil should be achieved for a given flight condition. An example for an optimal control task would be to keep a process temperature of a chemical reactor as close to a specified temperature as possible in a dynamically changing environment.

Note, that the inverse design problem can be reformulated as optimization problem, as it aims at minimizing the deviation between the current state of the output variables and the desired state.

In multi-objective optimization we look at the optimization of systems w.r.t. more than one output variables. Single-objective optimization can be considered as a special case of multi-objective optimization with only one output variable.

Moreover, classically, multi-objective optimization problems are most of the time reduced to single-objective optimization problems. We refer to these reduction techniques as *scalarization* techniques. A chapter in these lecture notes is

devoted to this topic. Modern techniques, however, often aim at obtaining a set of 'interesting' solutions by means of so-called Pareto optimization techniques. What is meant by this will be discussed in the remainder of this chapter.

1.2 Formal Problem Definitions in Mathematical Programming

Researchers in the field of *operations research* use an elegant and standardized notation for the classification and formalization of optimization and decision problems, the so-called *mathematical programming problems*, among which linear programs (LP) are certainly the most prominent representant. Using this notion a *generic definition of optimization problems* is as follows:

$$
\begin{aligned}
f(\mathbf{x}) &\rightarrow \min! && \text{(* Objectives *)} && (1.1)\\
g_1(\mathbf{x}) &\leq 0 && \text{(* Inequality constraints *)} && (1.2)\\
&\vdots && && (1.3)\\
g_{n_g}(\mathbf{x}) &\leq 0 && && (1.4)\\
h_1(\mathbf{x}) &= 0 && \text{(* Equality Constraints *)} && (1.5)\\
&\vdots && && (1.6)\\
h_{n_h}(\mathbf{x}) &= 0 && && (1.7)\\
\mathbf{x} &\in \mathcal{X} = [\mathbf{x}^{\min}, \mathbf{x}^{\max}] \subset \mathbb{R}^{n_x} \times \mathbb{Z}^{n_z} && \text{(* Box constraints *)} && (1.8)\\
& && && (1.9)
\end{aligned}
$$

The objective function f is a function to be minimized (or maximized[1]). This is the goal of the optimization. The function f can be evaluated for every point $\mathbf{x}$ in the search space (or decision space). Here the *search space* is defined by a set of intervals, that restrict the range of variables, so called bounds or box constraints. Besides this, variables can be integer variables that is they are chosen from $\mathbb{Z}$ or subsets of it, or continuous variable (from $\mathbb{R}$). An important special case of integer variables are binary variables which are often used to model binary decisions in mathematical programming.

Whenever inequality and equality constraints are stated explicitly, the search space $\mathcal{X}$ can be partitioned in a *feasible search space* $\mathcal{X}_f \subseteq \mathcal{X}$ and an *infeasible subspace* $\mathcal{X} - \mathcal{X}_f$. In the feasible subspace all conditions stated in the mathematical programming problem are satisfied. The conditions in the mathematical

[1]Maximization can be rewritten as minimization by changing the sign of the objective function, that is, replacing $f(\mathbf{x}) \rightarrow \max$ with $-f(\mathbf{x}) \rightarrow \min$

program are used to avoid constraint violations in the system under design, e.g., the excess of a critical temperature or pressure in a chemical reactor (an example for an inequality constraint), or the keeping of conservation of mass formulated as an equation (an example for an equality constraint). The conditions are called constraints. Due to a convention in the field of operations research, constraints are typically written in a *standardized form* such that 0 appears on the right hand side side. Equations can easily be transformed into the standard form by means of algebraic equivalence transformations.

Based on this very general problem definition we can define several classes of optimization problems by looking at the characteristics of the functions f, $g_i, i = 1, \ldots, n_g$, and $h_i, i = 1, \ldots, n_h$. Some important classes are listed in Table 1.1.

Name	Abbreviation	Search Space	Functions
Linear Program	LP	$\mathbb{R}^{n_r}$	linear
Quadratic Program	QP	$\mathbb{R}^{n_r}$	quadratic
Integer Linear Progam	ILP	$\mathbb{Z}^{n_z}$	linear
Integer Progam	IP	$\mathbb{Z}^{n_z}$	arbitrary
Mixed Integer Linear Program	MILP	$\mathbb{Z}^{n_z} \times \mathbb{R}^{n_r}$	linear
Mixed Integer Nonlinear Program	MINLP	$\mathbb{Z}^{n_z} \times \mathbb{R}^{n_r}$	nonlinear

Table 1.1: Classification of mathematical programming problems.

1.2.1 Other Problem Classes in Optimization

There are also other types of mathematical programming problems. These are, for instance based on:

- The handling of uncertainties and noise of the input variables and of the parameters of the objective function: Such problems fall into the class of *robust optimization problems* and *stochastic programming*. If some of the constants in the objective functions are modeled as stochastic variables the corresponding problems are also called a *parametric optimization problem*.

- Non-standard search spaces: Non-standard search spaces are for instance the search spaces of trees (e.g. representing regular expressions), network configurations (e.g. representing flowsheet designs) or searching for 3-D structures (e.g. representing bridge constructions). Such problem are referred to as *topological*, *grammatical*, or *structure optimization*.

- A finer distinction between different mathematical programming problems based on the characteristics of the functions: Often subclasses of mathematical programming problems have certain mathematical properties that

can be exploited for faster solving them. For instance *convex quadratic programming problems* form a special class of relatively easy to solve quadratic programming problems. Moreover, *geometrical programming problems* are an important subclass of nonlinear programming tasks with polynomials that are allowed to h negative numbers or fractions as exponents.

In some cases, the demand that a solution consists of a vector of variables is too restrictive and instead we can define the search space as some set $\mathcal{X}$. In order to capture also these kind of problems a more general definition of a general optimization problem can be used:

$$f_1(\mathbf{x}) \to \min, \quad \mathbf{x} \in \mathcal{X} \tag{1.10}$$

$\mathbf{x} \in \mathcal{X}$ is called the *search point* or *solution candidate* and $\mathcal{X}$ is the search space or decision space. Finally, $f : \mathcal{X} \to \mathbb{R}$ denotes the objective function. Only in cases where $\mathcal{X}$ is a vector space, we may talk of a *decision vector*.

Another important special case is given, if $\mathcal{X} = \mathbb{R}^n$. Such problems are defined as *continuous unconstrained optimization problems* or, simply, unconstrained optimization problems.

For notational convenience in the following we will refer mostly to the generic definition of an optimization problem given in equation 1.10, whenever constraint treatment is not particularly addressed. In such cases we assume that $\mathcal{X}$ already contains only feasible solutions.

1.2.2 Multiobjective Optimization

All optimization problems and mathematical programming problem classes can be generalized to multiobjective optimization problem classes by stating multiple objective functions:

$$f_1(\mathbf{x}) \to \min, \ldots, f_m(\mathbf{x}) \to \min, \quad \mathbf{x} \in \mathcal{X} \tag{1.11}$$

At this point in time it is not clear, how to deal with situations with conflicting objectives, e.g. when it is not possible to minimize all objective functions simultaneously. Note that the problem definition does not yet prescribe how to compare different solutions. To discuss this we will introduce concepts from the theory of ordered sets, such as the Pareto dominance relation. A major part of these lecture notes will then be devoted to the treatise of multiobjective optimization.

Before proceeding in this direction, it is however important to note, that many difficulties of solving single objective optimization problems are inherited by the more general class of multiobjective optimization problems. We will therefore first summarize these.

1.3 Problem Difficulty in Optimization

The way problem difficulty is defined in continuous unconstrained optimization differs widely from the concepts typically referred to in discrete optimization. This is why we look at these two classes separately. Thereafter we will show that discrete optimization problems can be formulated as constrained continuous optimization problems, or, referring to the classification scheme in Table 1.1, as nonlinear programming problems.

1.3.1 Problem Difficulty in Continuous Optimization

In continuous optimization the metaphor of a optimization landscape is often used in order to define problem difficulty. As opposed to just talking about a function, when using the term (search) *landscapes* one explicitly requires the search space to be equipped with a neighborhood structure, which could be a metric or a topology. This topology is typically the standard topology on $\mathbb{R}^n$ and as a metric typically the Euclidean metric is used.

As we will discuss in more rigor in Chapter 4, this gives rise to definitions such as *local optima*, which are points that cannot be improved by replacing them by neighboring points. For many optimum seeking algorithms it is difficult to escape from such points or find a good direction (in case of plateaus). If local optima are not also global optima the algorithm might return a suboptimal solutions.

Problems with multiple local optima are called *multimodal optimization problems*, whereas a problem that has only a single local optimum is called a *unimodal optimization problem*. Multimodal optimization problems are, in general, considered to be more difficult to solve than unimodal optimization problems. However, in some cases unimodal optimization problems can be very difficult, too. For instance, in case of large neighbourhoods it can be hard to find the neighbour that improves a point.

Exampes of continuous optimization problems are given in Figure . The problem TP2 is difficult due to discontinuities. The function TP3 has only one local optimum (unimodal) and no discontinuities; therefore it can be expected that local optimization can easily solve this problem. Highly multimodal problems are given in TP5 and TP6.

Another difficulty is imposed by constraints. In constrained optimization problems optima can be located at the boundary of the search space and they can give rise to disconnected feasible subspaces. Again, connectedness is a property that requires the definition of neighbourhoods. The definition of a continuous path can be based on this, which again is used to define connectivity. The reason why disconnected subspaces make problems hard to solve are, similar to the multimodal case, that in these problems barriers are introduced might prevent optimum seeking

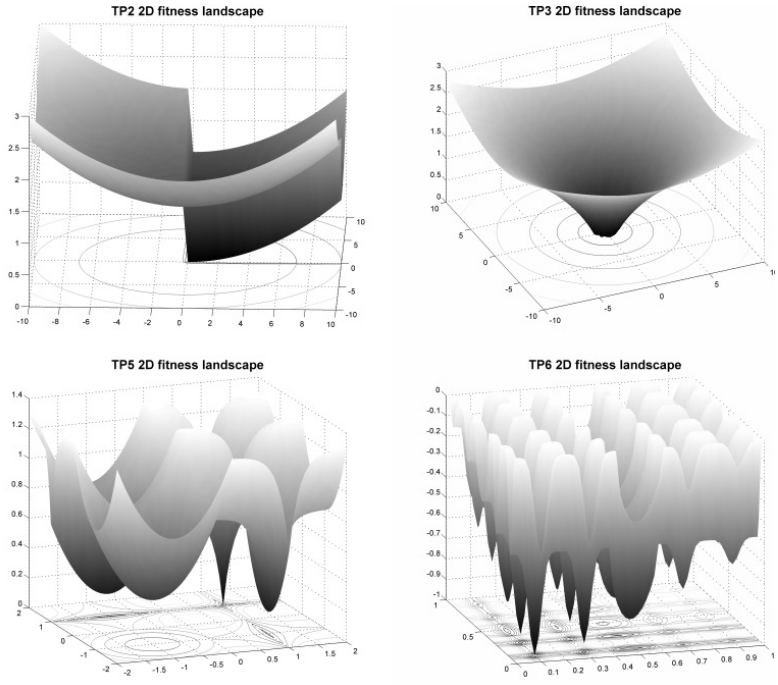

Figure 1.2: Examples of continuous optimization problems.

algorithms that use strategies of gradual improvement to find the global optimum.

Finally, discontinuities and ruggedness of the landscape make problems difficult to solve for many solvers. Discontinuities are abrupt changes of the function value in some neighborhood. In particular these cause difficulties for optimization methods that assume the objective function to be continuous, that is they assume that similar inputs cause similar outputs. A common definition of continuity is that of Lipschitz continuity:

Definition 1 *Let $d(x, y)$ denote the Euclidean distance between two points in the search space. Then function f is Lipschitz continuous, if and only if*

$$|f(x) - f(y)| < kd(x, y) \text{ for some } k > 0.$$

For instance the work by Ritter and Novak [66] has clarified that Lipschitz continuity alone is not sufficient to guarantee that a problem is easy to solve. However, continuity can be exploited for guaranteeing that a region has been sufficiently explored and therefore a small Lipschitz constant has a dampening effect on the worst case time complexity for continuous global optimization, which, even given Lipschitz continuity, grows exponentially with the number of variables involved [66]. In cases where we omit continuity assumptions the time complexity might

even grow super-exponentially. Here complexity is defined as the number of function evaluations it takes to get a solution that has a distance of ϵ to the global optimum and it is assumed that the variables are restricted in an closed interval range.

As indicated above, the number of optimization variables is another source of difficulty in continuous optimization problems. In particular, if f is a black box function it is known that even for Lipschitz continuous problems the number of required function evaluations for finding a good approximation to the global optimum grows exponentially with the number of decision variables. This result is also referred to as the *curse of dimensionality*.

Again, a word of caution is in order: The fact that a problem is low dimensional or even one dimensional in isolation does not say something about its complexity. *Kolmogorov's superposition theorem* shows that every continuous multivariate function can be represented by a one dimensional function, and it is therefore often possible to re-write optimization problems with multiple variables as one-dimensional optimization problems.

Besides continuity assumptions, also differentiability of the objective function and constraints, convexity and mild forms of non-linearity (as given in convex quadratic optimization), as well as limited interaction between variables can make a continuous problem easier to solve. The degree of interaction between variables is given by the number of variables in the term of the objectve function: Assume it is possible to (re)-write the optimization problem in the form $\sum_{i=1}^{n} f_i(x_{i_1}, \ldots, x_{i_{k(i)}}) \rightarrow \max$, then the value of $k(i)$ is the degree of interaction in the i-th component of the objective function. In case of continuous objective functions it can be shown that problems with a low degree of interaction can be solved more efficiently in terms of worst case time complexity[66]. One of the reasons why convex quadratic problems can be solved efficiently is that, given the Hessian matrix, the coordinate system can be transformed by simple rotation in such a way that these problems become decomposable, i.e. $k(i)$ is bounded by 1.

1.3.2 Problem Difficulty in Combinatorial Optimization

Many optimization problem in practice, such as scheduling problems, subset selection problems, and routing problems, belong to the class of *combinatorial optimization problems* and, as the name suggests, they look in some sense for the best combination of parts in a solution (e.g. selected elements of a set, travelled edges in a road network, switch positions in a boolean network). Combinatorial optimization problems are problems formulated on (large) finite search spaces. In the classification scheme in Table 1.1 they belong to the classses IP and ILP. Although combinatorial optimization problems are originally not always formulated on search spaces with integer decision variables, most combinatorial optimization

problems can be transformed to equivalent IP and ILP formulations with binary decision variables. For the sake of brevity, the following discussion will focus on binary unconstrained problems. Most constrained optimization problems can be transformed to equivalent unconstrained optimization problems by simply assigning a sufficiently large ('bad') objective function value to all infeasible solutions.

A common characteristic of many combinatorial optimization problems is that they have a concise (closed form) formulation of the objective function and the objective function (and the constraint functions) can be computed efficiently.

Having said this, a combinatorial optimization problem can be defined by means of a pseudo-boolean objective function, i.e. $f : \{0,1\}^n \rightarrow \mathbb{R}$ and stating the goal $f(\mathbf{x}) \rightarrow \min$. Theoretical computer science has developed a rich theory on the complexity of decision problems. A *decision problem* is the problem of answering a query on input of size n with the answer being either `yes` or `no`. In order to relate the difficulty of optimization problems to the difficulty of decision problems it is beneficial to formulate so-called decision versions of optimization problems.

Definition 2 *Given an combinatorial optimization problem of the form* $f(\mathbf{x}) \rightarrow \max$ *for* $\mathbf{x} \in \{0,1\}^n$ *its* decision version *is defined as the query:*

$$\exists \mathbf{x} \in \{0,1\}^n : f(\mathbf{x}) \leq k \tag{1.12}$$

for a given value of $k \in \mathbb{R}$.

NP hard combinatorial optimization problems

A decision problem is said to belong to the class P if there exists an algorithm on a Turing machine[2] that solves it with a time complexity that grows at most polynomially with the size n of the input. It belongs to the class NP if a candidate solution $\mathbf{x}$ of size n can be verified ('checked') with polynomial time complexity (does it satify the formula $f(\mathbf{x}) \leq k$ or not). Obviously, the class NP subsumes the class P, but the P perhaps not neccessarily subsumes NP. In fact, the question whether P subsumes NP is the often discussed open problem in theoretical computer science known as the '$P = NP$' problem. Under the assumption '$P \neq NP$', that is that P does include NP, it is meaningful to introduce the complexity class of NP complete problems:

Definition 3 *A decision problem* D *is* NP complete*, if and only if for all problems* D' *in* NP *there exists an algorithm with polynomial time complexity that*

[2]or any in any common programming language operating on infinite memory and not using parallel processing and not assuming constant time complexity for certain infinite precision floating point operations such as the floor function.

reformulates an instance of D as an instance of D' (One can also say: 'there exists a polynomial-time reduction of D to P').

If any NP complete problem could be solved with polynomial time complexity then all problems in NP have polynomial time complexity.

Many decision versions of optimization problems are NP complete. Closely related to the class of NP complete problems is the class of NP hard problems.

Definition 4 (NP hard) *A computational problem is NP hard, if and only if any problem in the class of NP complete problems can in polynomial time be reduced to this problem.*

That a problem is NP hard does not imply that it is in NP. Moreover, given any NP hard problem could be solved in polynomial time, then all problems in NP could be solved in polynomial time, but not vice versa.

Many combinatorial optimization problems fall into the class of NP hard problems and their decision versions belong to the class of NP complete problems. Examples of NP hard optimization problems are the knapsack problem, the traveling salesperson problem, and integer linear programming (ILP).

At this point in time, despite considerable efforts of researchers, no polynomial time algorithms are known for NP complete problems and thus also not for NP hard problems. As a consequence, relying on currently known algorithms the computational effort to solve NP complete (NP hard) problems grows (at least) exponentially with the size n of the input.

The fact that a given problem instance belongs to a class of NP complete problems does not mean that this instance itself is hard to solve. Firstly, the exponential growth is a statement about the *worst case* time complexity and thus gives an upper bound for the time complexity that holds for all instances of the class. It might well be the case that for a given instance the worst case is not binding. Often certain structural features such as *bounded tree width* reveal that an instance belongs to an easier to solve subclass of a NP complete problem. Moreover, exponential growth might occur with a small growth rate and problem sizes relevant in practice might still be solvable in an acceptable time.

Continuous vs. discrete optimization

Given that some continuous versions of mathematical programming problems belong to easier to solve problem classes than their discrete counterparts one might ask the question whether integer problems are essentially more difficult to solve than continuous problems.

As a matter of fact, optimization problems on binary input spaces can be

reformulated as quadratic optimization problems by means of the following construction:

Given an integer programming problem with binary decision variables $b_i \in \{0, 1\}$, $i = 1, \ldots, n$, we can reformulate this problem as a quadratic programming problem with continuous decision variables $x_i \in \mathbb{R}$ by introducing the constraints $(x_i)(1 - x_i) = 1$ for $i = 1, \ldots, n$.

For obvious reasons continuous optimization problems cannot always be formulated as discrete optimization problems. However, it is sometimes argued that all problems solved on digital computers are discrete problems and infinite accuracy is almost never required in practise. If, however, infinite accuracy operations are assumed to have constant time in the construction of an algorithm this can lead to strange consequences. For instance, it is known that polynomial time algorithms can be constructed for NP complete problems, if one would accept that the evaluation of the floor function with infinite precision can be achieved in polynomial time. However, it is not possible to implement these algorithms on a von Neumann architecture with finite precision arithmetics.

Finally, in times of growing amounts of decision data, one should not forget that even guarantees of polynomial time complexity can be insufficient in practise. Accordingly, there is a growing interest for problem solvers that require only subquadratic running time. Similar to the construction of the class of NP complete problems, a definition of the class of $3SUM$ complete problems has been constructed by theoretical computer scientists. For this class up to date only quadratic running time algorithms are known. A prominent problem from the domain of mathematical programming that belongs to this group is the *linear satisfiability problem*, i.e. the problem of whether a set of r linear inequality constraints formulated on n continuous variables can be satisfied [33].

1.4 Pareto dominance

A fundamental problem in multicriteria optimization and decision making is to compare solutions w.r.t. different, possibly conflicting, goals. Before we lay out the theory of orders in a more rigorous manner, we will introduce some fundamental concepts by means of a simple example.

Consider the following decision problem: We have to select one car from the following set of cars: For the moment, let us assume, that our goal is to minimize the price and maximize speed and we do not care about other components.

In that case we can clearly say that the BMW outperforms the Lincoln stretch limousine, which is at the same time more expensive and slower then the BMW. In such a situation we can decide clearly for the BMW. We say that the first solution *(Pareto) dominates* the second solution. Note, that the concept of Pareto

Criterion	Price [kEuro]	Maximum Speed [km/h]	length [m]	color
VW Beetle	3	120	3.5	red
Ferrari	100	232	5	red
BMW	50	210	3.5	silver
Lincoln	60	130	8	white

domination is named after Vilfredo Pareto, an italian economist and engineer who lived from 1848-1923 and who introduced this concept for multi-objective comparisons.

Consider now the case, that you have to compare the BMW to the VW Beetle. In this case it is not clear how to make a decision, as the beetle outperforms the BMW in the cost objective, while the BMW outperforms the VW Beetle in the speed objective. We say that the two solutions are *incomparable*. Incomparability is a very common characteristic that occurs in so-called *partial ordered* sets.

We can also observe, that the BMW is incomparable to the Ferrari, and the Ferrari is incomparable to the VW Beetle. We say these three cars form a set of mutually incomparable solutions. Moreover, we may state that the Ferrari is incomparable to the Lincoln, and the VW Beetle is incomparable to the Lincoln. Accordingly, also the VW Beetle, the Lincoln and the Ferrari form a mutually incomparable set.

Another characteristic of a solution in a set can be that it is *non-dominated* or Pareto optimal. This means that there is no other solution in the set which dominates it. The set of all non-dominated solutions is called the *Pareto front*. It might exist of only one solution (in case of non-conflicting objectives) or it can even include no solution at all (this holds only for some infinite sets). Moreover, the Pareto set is always a mutually incomparable set. In the example this set is given by the VW Beetle, the Ferrari, and the BMW.

An important task in multi-objective optimization is to identify the Pareto front. Usually, if the number of objective is small and there are many alternatives, this reduces the set of alternatives already significantly. However, once the Pareto front has been obtained, a final decision has to be made. This decision is usually made by interactive procedures where the decision maker assesses trade-offs and sharpens constraints on the range of the objectives. In the subsequent chapters we will discuss these procedures in more detail.

Turning back to the example, we will now play a little with the definitions and thereby get a first impression about the rich structure of partially ordered sets in Pareto optimization: What happens if we add a further objective to the set of objectives in the car-example? For example let us assume, we also would like to have a very big car and the size of the car is measured by its length! It is easy to verify that the size of the non-dominated set increases, as now the Lincoln is

also incomparable to all other cars and thus belongs to the non-dominated set. Later we will prove that introducing new objectives will always increase the size of the Pareto front. On the other hand we may define a constraint that we do not want a silver car. In this case the Lincoln enters the Pareto front, since the only solution that dominates it leaves the set of feasible alternatives. In general, the introduction of constraints may increase or decrease Pareto optimal solutions or its size remains the same.

1.5 Formal Definition of Pareto Dominance

A formal and precise definition of Pareto dominance is given as follows. We define a partial order[3] on the *solution space* $\mathcal{Y} = f(\mathcal{X})$ by means of the Pareto dominance concept for vectors in $\mathbb{R}^m$:

For any $\mathbf{y}^{(1)} \in \mathbb{R}^m$ and $\mathbf{y}^{(2)} \in \mathbb{R}^m$: $\mathbf{y^{(1)}}$ dominates $\mathbf{y^{(2)}}$ (in symbols $\mathbf{y^{(1)}} \prec_{Pareto} \mathbf{y^{(2)}}$) if and only if: $\forall i = 1, \ldots m : y_i^{(1)} \leq y_i^{(2)}$ and $\exists i \in \{1, \ldots, m\} : y_i^{(1)} < y_i^{(2)}$.

Note, that in the bi-criteria case this definition reduces to: $\mathbf{y}^1 \prec_{Pareto} \mathbf{y}^2 :\Leftrightarrow y_1^{(1)} < y_1^{(2)} \wedge y_2^{(1)} \leq y_2^{(2)} \vee y_1^{(1)} \leq y_1^{(2)} \wedge y_2^{(1)} < y_2^{(2)}$.

In addition to the domination $\prec_{Pareto}$ we define further comparison operators: $\mathbf{y}^{(1)} \preceq_{Pareto} \mathbf{y}^{(2)} :\Leftrightarrow \mathbf{y}^{(1)} \prec_{Pareto} \mathbf{y}^{(2)} \vee \mathbf{y}^{(1)} = \mathbf{y}^{(2)}$.

Moreover, we say $\mathbf{y}^{(1)}$ is incomparable to $\mathbf{y}^{(2)}$ (in symbols: $\mathbf{y}^{(1)} || \mathbf{y}^{(2)}$),if and only if $\mathbf{y}^{(1)} \not\preceq_{Pareto} \mathbf{y}^{(2)} \wedge \mathbf{y}^{(1)} \not\preceq_{Pareto} \mathbf{y}^{(2)}$.

For technical reasons, we also define *strict* domination as: $\mathbf{y}^{(1)}$ strictly dominates $\mathbf{y}^{(2)}$, iff $\forall i = 1, \ldots, m : y_i^{(1)} < y_i^{(2)}$.

For any compact subset of R^m, say $\mathcal{Y}$, there exists a non-empty set of minimal elements w.r.t. the partial order $\preceq$ (cf. [Ehr05, page 29]). Minimal elements of this partial order are called non-dominated points. Formally, we can define a non-dominated set via: $\mathcal{Y}_N = \{\mathbf{y} \in \mathcal{Y} | \nexists \mathbf{y}' \in \mathbf{Y} : \mathbf{y}' \prec_{\mathbf{Pareto}} \mathbf{y}\}$. Following a convention by Ehrgott [Ehr05] we use the index N to distinguish between the original set and its non-dominated subset.

Having defined the non-dominated set and the concept of Pareto domination for general sets of vectors in $\mathbb{R}^m$, we can now relate it to the optimization task: The aim of Pareto optimization is to find the non-dominated set $\mathcal{Y}_N$ for $\mathcal{Y} = f(\mathcal{X})$ the image of $\mathcal{X}$ under f, the so-called *Pareto front* of the multi-objective optimization problem.

We define $\mathcal{X}_E$ as the inverse image of $\mathcal{Y}_N$, i. e. $\mathcal{X}_E = f^{-1}(\mathcal{Y}_N)$. This set will be called the *efficient set* of the optimization problem. Its members are called *efficient solutions*.

[3]Partial orders will be defined in detail in Chapter 2. For now, we can assume that it is an order where not all elements can be compared.

For notational convenience, we will also introduce an order (which we call prePareto) on the decision space via $\mathbf{x}^{(1)} \prec_{prePareto} \mathbf{x}^{(2)} \Leftrightarrow f(\mathbf{x}^{(1)}) \prec_{Pareto} f(\mathbf{x}^{(2)})$. Accordingly, we define $\mathbf{x}^{(1)} \preceq_{prePareto} \mathbf{x}^{(2)} \Leftrightarrow f(\mathbf{x}^{(1)}) \preceq_{Pareto} f(\mathbf{x}^2)$. Note, the minimal elements of this order are the efficient solutions, and the set of all minimal elements is equal to $\mathcal{X}_E$.

Exercises

1. How does the introduction of a new solution influence the size of the Pareto set? What happens if solutions are deleted? Prove your results!

2. Why are objective functions and constraint functions essentially different? Give examples of typical constraints and typical objectives in real world problems!

3. Find examples for decision problems with multiple, conflicting objectives! How is the search space defined? What are the constraints, what are the objectives? How do these problems classify, w.r.t. the classification scheme of mathematical programming? What are the human-centric aspects of these problems?

Part I

Foundations

Chapter 2

Orders and dominance

The theory of ordered sets is an essential analytical tool in multi-objective optimization and decision analysis. Different types of orders can be defined by means of axioms on binary relations, and, if we restrict ourselves to vector spaces, also geometrically.

Next we will first show how different types of orders are defined as binary relations that satisfy a certain axioms[1]. Moreover, we will highlight the essential differences between common families of ordered sets, such as preorders, partial orders, linear orders, and cone orders.

The structure of this chapter is as follows: After reviewing the basic concept of binary relations, we define some axiomatic properties of pre-ordered sets, a very general type of ordered sets. Then we define partial orders and linear orders as special type of pre-orders. The difference between linear orders and partial orders sheds a new light on the concept of incomparability and the difference between multicriteria and single criterion optimization. Later, we discuss techniques how to visualize finite ordered sets in a compact way, by so called Hasse diagrams. The remainder of this chapter deals with an alternative way of defining orders on vector spaces: Here we define orders by means of cones. This definition leads also to an intuitive way of how to visualize orders based on the concept of Pareto domination.

2.1 Preorders

Orders can be introduced and compared in an elegant manner as binary relations that obey certain axioms. Let us first review the definition of a binary relation

[1]Using here the term 'axiom' to refer to an elementary statement that is used to define a class of objects (as promoted by, for instance, Rudolf Carnap[16]) rather than viewing them as self-evident laws that do not require proof (Euclid's classical view).

and some common axioms that can be introduced to specify special subclasses of binary relations and that are relevant in the context of ordered sets.

Definition 5 *A* binary relation $\mathcal{R}$ *on some set* $\mathcal{S}$ *is defined as a set of pairs of elements of* $\mathcal{S}$*, that is, a subset of* $\mathcal{S} \times \mathcal{S} = \{(\mathbf{x}^1, \mathbf{x}^2) | \, \mathbf{x}^1 \in \mathcal{S} \text{ and } \mathbf{x}^2 \in \mathcal{S}\}$*. We write* $\mathbf{x}^1 \mathcal{R} \mathbf{x}^2 \Leftrightarrow (\mathbf{x}^1, \mathbf{x}^2) \in \mathcal{R}$*.*

Definition 6 *Properties of binary relations*

$\mathcal{R}$ *is* reflexive $\Leftrightarrow \forall \mathbf{x} \in \mathcal{S} : \mathbf{x} \mathcal{R} \mathbf{x}$

$\mathcal{R}$ *is* irreflexive $\Leftrightarrow \forall \mathbf{x} \in \mathcal{S} : \neg \mathbf{x} \mathcal{R} \mathbf{x}$

$\mathcal{R}$ *is* symmetric $\Leftrightarrow \forall \mathbf{x}^1, \mathbf{x}^2 \in \mathcal{S} : \mathbf{x}^1 \mathcal{R} \mathbf{x}^2 \Leftrightarrow \mathbf{x}^2 \mathcal{R} \mathbf{x}^1$

$\mathcal{R}$ *is* antisymmetric $\Leftrightarrow \forall \mathbf{x}^1, \mathbf{x}^2 \in \mathcal{S} : \mathbf{x}^1 \mathcal{R} \mathbf{x}^2 \wedge \mathbf{x}^2 \mathcal{R} \mathbf{x}^1 \Rightarrow \mathbf{x}^1 = \mathbf{x}^2$

$\mathcal{R}$ *is* asymmetric $\Leftrightarrow \forall \mathbf{x}^1, \mathbf{x}^2 \in \mathcal{S} : \mathbf{x}^1 \mathcal{R} \mathbf{x}^2 \Rightarrow \neg (\mathbf{x}^2 \mathcal{R} \mathbf{x}^1)$

$\mathcal{R}$ *is* transitive $\Leftrightarrow \forall \mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3 \in \mathcal{S} : \mathbf{x}^1 \mathcal{R} \mathbf{x}^2 \wedge \mathbf{x}^2 \mathcal{R} \mathbf{x}^3 \Rightarrow \mathbf{x}^1 \mathcal{R} \mathbf{x}^3$

Example It is worthwhile to practise these definitions by finding examples for structures that satisfy the aforementioned axioms. An example for a reflexive relation is the equality relation on $\mathbb{R}$, but also the relation $\leq$ on $\mathbb{R}$. A classical example for a irreflexive binary relation would be marriage between two persons. This relation is also symmetric. Symmetry is also typically a characteristic of neighborhood relations – if A is neighbor to B then B is also neighbor to A.

Antisymmetry is exhibited by $\leq$, the standard order on $\mathbb{R}$, as $x \leq y$ and $y \leq x$ entails $x = y$. Relations can be at the same time symmetric and antisymmetric: An example is the equality relation. Antisymmetry will also occur in the axiomatic definition of a partial order, discussed later. Asymmetry, not to be confused with antisymmetry, is somehow the counterpart of symmetry. It is also a typical characteristic of strictly ordered sets – for instance $<$ on $\mathbb{R}$.

An example of a binary relation (which is not an order) that obeys the transitivity axiom is the path-accessibility relation in directed graphs. If node B can be reached from node A via a path, and node C can reached from node B via a path, then also node C can be reached from node A via a path.

2.2 Preorders

Next we will introduce preorders and some properties on them. Preorders are a very general type of orders. Partial orders and linear orders are preorders that obey additional axioms. Beside other reasons these types of orders are important, because the Pareto order used in optimization defines a partial order on the objective space and a pre-order on the search space.

Definition 7 *Preorder*
A preorder *(quasi-order) is a binary relation that is both transitive and reflexive. We write* $\mathbf{x}^1 \preceq_{pre} \mathbf{x}^2$ *as shorthand for* $\mathbf{x}^1 \mathcal{R} \mathbf{x}^2$*. We call* $(\mathcal{S}, \preceq_{pre})$ *a* preordered set*.*

In the sequel we use the terms preorder and order interchangeably. Closely related to this definition are the following derived notions:

Definition 8 *Strict preference*
$\mathbf{x}^1 \prec_{pre} \mathbf{x}^2 :\Leftrightarrow \mathbf{x}^1 \preceq_{pre} \mathbf{x}^2 \wedge \neg(\mathbf{x}^2 \preceq_{pre} \mathbf{x}^1)$

Definition 9 *Indifference*
$\mathbf{x}^1 \sim_{pre} \mathbf{x}^2 :\Leftrightarrow \mathbf{x}^1 \preceq_{pre} \mathbf{x}^2 \wedge \mathbf{x}^2 \preceq_{pre} \mathbf{x}^1$

Definition 10 *Incomparability*
A pair of solutions $\mathbf{x}^1, \mathbf{x}^2 \in \mathcal{S}$ *is said to be incomparable, iff neither* $\mathbf{x}^1 \preceq_{pre} \mathbf{x}^2$ *nor* $\mathbf{x}^2 \preceq_{pre} \mathbf{x}^1$*. We write* $\mathbf{x}^1 || \mathbf{x}^2$*.*

Strict preference is irreflexive and transitive, and, as a consequence asymmetric. Indifference is reflexive, transitive, and symmetric. The properties of the incomparability relation we leave for exercise.

Having discussed binary relations in the context of pre-orders, let us now turn to characteristics of pre-ordered sets. One important characteristic of pre-orders in the context of optimization is that they are elementary structures on which minimal and maximal elements can be defined. Minimal elements of a pre-ordered set are elements that are not preceded by any other element.

Definition 11 *Minimal and maximal elements of an pre-ordered set* $\mathcal{S}$
$\mathbf{x}^1 \in \mathcal{S}$ *is minimal, if and only if not exists* $\mathbf{x}^2 \in \mathcal{S}$ *such that* $\mathbf{x}^2 \prec_{pre} \mathbf{x}^1$
$\mathbf{x}^1 \in \mathcal{S}$ *is maximal, if and only if not exists* $\mathbf{x}^2 \in \mathcal{S}$ *such that* $\mathbf{x}^1 \prec_{pre} \mathbf{x}^2$

Proposition 12 *For every finite set (excluding here the empty set* $\emptyset$*) there exists at least one minimal and at least one maximal element.*

For infinite sets, pre-orders with infinite many minimal (maximal) elements can be defined and also sets with no minimal (maximal) elements at all, such as the natural numbers with the order $<$ defined on them, for which there exists no maximal element. Turning the argument around, one could elegantly define an infinite set as a non-empty set on which there exists a pre-order that has no maximal element.

In absence of any additional information the number of pairwise comparisons required to find all minimal (or maximal) elements of a finite pre-ordered set of size $|\mathcal{X}| = n$ is $\binom{n}{2} = \frac{(n-1)n}{2}$. This follows from the effort required to find the minima in the special case where all elements are mutually incomparable.

2.3 Partial orders

Pareto domination imposes a partial order on a set of criterion vectors. The definition of a partial order is more strict than that of a pre-order:

Definition 13 *Partial order*
A **partial order** *is a preorder that is also* antisymmetric. *We call* $(\mathcal{S}, \preceq_{partial})$ *a partially ordered set* or **poset**.

As partial orders are a specialization of preorders, we can define *strict preference* and *indifference* as before. Note, that for partial orders two elements that are indifferent to each other are always equal: $\mathbf{x}^1 \sim \mathbf{x}^2 \Rightarrow \mathbf{x}^1 = \mathbf{x}^2$

To better understand the difference between pre-ordered sets and posets let us illustrate it by means of two examples:

Example
A pre-ordered set that is not a partially ordered set is the set of complex numbers $\mathbb{C}$ with the following precedence relation:

$$\forall (z_1, z_2) \in \mathbb{C}^2 : z_1 \preceq z_2 :\Leftrightarrow |z_1| \leq |z_2|.$$

It is easy to verify reflexivity and transitivity of this relation. Hence, $\preceq$ defines a pre-order on $\mathbb{C}$. However, we can easily find an example that proves that antisymmetry does not hold. Consider two distinct complex numbers $z = -1$ and $z' = 1$ on the unit sphere (i.e. with $|z| = |z'| = 1$. In this case $z \preceq z'$ and $z' \preceq z$ but $z \neq z'$ ∎

Example
An example for a partially ordered set is the subset relation $\subseteq$ on the power set[2] $\wp(S)$ of some finite set S. Reflexivity is given as $A \subseteq A$ for all $A \in \wp(S)$. Transitivity is fulfilled, because $A \subseteq B$ and $B \subseteq C$ implies $A \subseteq C$, for all triples (A, B, C) in $\wp(S) \times \wp(S) \times \wp(S)$. Finally, antisymmetry is fulfilled, since $A \subseteq B$ and $B \subseteq A$ implies $A = B$ for all pairs $(A, B) \in \wp(S) \times \wp(S)$ ∎

Remark In general the Pareto order on the search space is a preorder which is not always a partial order in contrast to the Pareto order defined on the objective space (that is, the Pareto order is always a partial order).

[2] the power set of a set is the set of all subsets including the empty set

2.4 Linear orders and anti-chains

Perhaps the most well-known specializations of a partially ordered sets are linear orders. Examples for linear orders are the $\leq$ relations on the set of real numbers or integers. These types of orders play an important role in single criterion optimization, while in the more general case of multiobjective optimization we deal typically with partial orders that are not linear orders.

Definition 14 (Linear order) *A* **linear** *(or:***total) order** *is a partial order that satisfies also the* comparability *or* totality *axiom:* $\forall \mathbf{x}^1, \mathbf{x}^2 :\in \mathcal{X} : \mathbf{x}^1 \preceq \mathbf{x}^2 \vee \mathbf{x}^2 \preceq \mathbf{x}^1$

Totality is only axiom that distinguishes partial orders from linear orders. This also explains the name 'partial' order. The 'partiality' essentially refers to the fact that not all elements in a set can be compared, and thus, as opposed to linear orders, there are incomparable pairs.

A linearly ordered set is also called a (also called *chain*). The counterpart of the chain is the anti-chain:

Definition 15 (Anti-chain) *A poset* $(\mathcal{S}, \preceq_{partial})$ *is said to be an* **antichain***, iff:* $\forall \mathbf{x}^1, \mathbf{x}^2 \in \mathcal{S} : \mathbf{x}^1 || \mathbf{x}^2$

When looking at sets on which a Pareto dominance relation $\preceq$ is defined, we encounter subsets that can be classified as anti-chains and subsets that can be classified as linear orders, or non of these two. Examples of anti-chains are *Pareto fronts*.

Subsets of ordered sets that form anti-chain play an important role in characterizing the time complexity when searching for minimal elements, as the following recent result shows [21]:

Theorem 16 (Finding minima of bounded width posets) *Given a poset* $(\mathcal{X}, \preceq_{partial})$*, then its* width w *is defined the maximal size of a mutually non-dominated subset. Finding the minimal elements of a poset of size* n *and width of size* w *has a time complexity in* $\Theta(wn)$ *and an algorithm has been specified that has this time complexity.*

In [21] a proof for this theorem is provided and efficient algorithms.

2.5 Hasse diagrams

One of the most attractive features of pre-ordered sets, and thus also for partially ordered sets is that they can be graphically represented. This is commonly done by so-called Hasse diagrams, named after the mathematician Helmut Hasse (1898 -

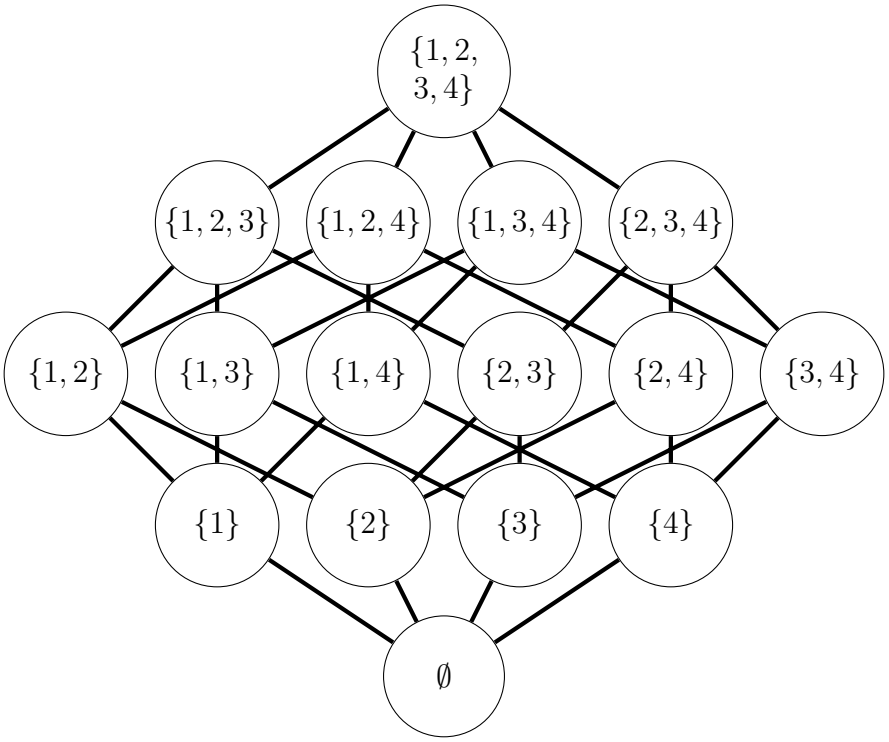

Figure 2.1: The Hasse Diagram for the set of all non-empty subsets partially ordered by means of $\subseteq$.

1979). The advantage of these diagrams, as compared to the graph representation of binary relations is essentially that edges that can be deduced by transitivity are omitted.

For the purpose of description we need to introduce the **covers** relation:

Definition 17 (Covers relation) *Given two elements* $\mathbf{x}^1$ *and* $\mathbf{x}^1$ *from a poset* $(\mathcal{X}, \prec_{partial})$. *Then* $\mathbf{x}^2$ covers $\mathbf{x}^1$, *in symbols* $\mathbf{x}^1 \lhd \mathbf{x}^2 :\Leftrightarrow \mathbf{x}^1 \prec_{partial} \mathbf{x}^2$ *and* $\mathbf{x}^1 \preceq_{partial} \mathbf{x}^3 \prec_{partial} \mathbf{x}^2$ *implies* $\mathbf{x}^1 = \mathbf{x}^3$.

One may also define the covers relation in more informal terms as: $\mathbf{x^2}$ covers $\mathbf{x^1}$ if and only if no element lies strictly between $\mathbf{x^1}$ and $\mathbf{x^2}$.

As an example, consider the covers relation on the linearly ordered set $(\mathbb{N}, \leq)$. Here $\mathbf{x}^1 \lhd \mathbf{x}^2$, iff $\mathbf{x}^2 = \mathbf{x}^1 + 1$. Note, that for $(\mathbb{R}, \leq)$ the covers relation is the empty set.

Another example where the covers relation has a simple interpretation is the subset relation $\subseteq$. In this example a set A is covered by a set B, if and only if B contains one additional element. In Fig. 2.1 the subset relation is summarized

in a Hasse diagram. In this diagram the cover relation defines the arcs. A good description of the algorithm to draw a Hasse diagram has been provided by Davey and Priestly ([22], page 11):

Algorithm 1 Drawing the Hasse Diagram

1: To each point $\mathbf{x} \in S$ assign a point $p(\mathbf{x})$, depicted by a small circle with centre $p(\mathbf{x})$
2: For each covering pair $\mathbf{x}^1$ and $\mathbf{x}^2$ draw a line segment $\ell(\mathbf{x}^1, \mathbf{x}^2)$.
3: Choose the center of circles in a way such that:
4: whenever $\mathbf{x}^1 \lhd \mathbf{x}^2$, then $p(\mathbf{x}^1)$ is positioned below $p(\mathbf{x}^2)$.
5: if $\mathbf{x}^3 \neq \mathbf{x}^1$ and $\mathbf{x}^3 \neq \mathbf{x}^2$, then the circle of $\mathbf{x}^3$ does not intersect the line segment $\ell(\mathbf{x}^1, \mathbf{x}^2)$

There are many ways of how to draw a Hasse diagram for a given order. Davey and Priestly [22] note that diagram-drawing is 'as much an science as an art'. Good diagrams should provide an intuition for symmetries and regularities, and avoid crossing edges.

2.6 Comparing ordered sets

(Pre)ordered sets can be compared directly and on a structural level. Consider the four orderings depicted in the Hasse diagrams of Fig. 2.2. It should be immediately clear, that the first two orders $(\preceq_1, \preceq_2)$ on X have the same structure, but they arrange elements in a different way, while orders $\preceq_1$ and $\preceq_3$ also differ in their structure. Moreover, it is evident that all comparisons defined in $\prec_1$ are also defined in $\prec_3$, but not vice versa (e.g. c and b are incomparable in $\preceq_1$). The ordered set on $\preceq_3$ is an *extension* of the ordered set $\preceq_1$. Another extension of $\preceq_1$ is given with $\preceq_4$.

Let us now define these concepts formally:

Definition 18 (Order equality) *An ordered set $(X, \preceq)$ is said to be* equal *to an ordered set $(X, \preceq')$, iff $\forall x, y \in X : x \preceq y \Leftrightarrow x \preceq' y$.*

Definition 19 (Order isomorphism) *An ordered set $(X', \prec')$ is said to be an* isomorphic *to an ordered set $(X, \preceq)$, iff there exists a mapping $\phi : X \to X'$ such that $\forall x, x' \in X : x \preceq x' \Leftrightarrow \phi(x) \preceq' \phi(x')$. In case of two isomorphic orders, a mapping ϕ is said to be an* order embedding map *or* order isomorphism.

Definition 20 (Order extension) *An ordered set $(X, \prec')$ is said to be an* extension *of an ordered set $(X, \prec)$, iff $\forall x, x' \in X : x \prec x' \Rightarrow x \prec' x'$. In the latter*

case, $\prec'$ is said to be compatible with $\prec$. A linear extension *is an extension that is totally ordered.*

Linear extensions play a vital role in the theory of multi-objective optimization. For Pareto orders on continuous vector spaces linear extensions can be easily obtained by means of any weighted sum scalarization with positive weights. In general, topological sorting can serve as a means to obtain linear extensions. Both topics will be dealt with in more detail later in this work. For now, it should be clear that there can be many extensions of the same order, as in the example of Fig. 2.2, where $(X, \preceq_3)$ and $(X, \preceq_4)$ are both (linear) extensions of $(X, \preceq_1)$.

Apart from extensions, one may also ask if the structure of an ordered set is contained as a substructure of another ordered set.

Definition 21 *Given two ordered sets $(X, \preceq)$ and $(X', \preceq')$. A map $\phi : X \to X'$ is called* order preserving*, iff $\forall x, x' \in X : x \preceq x' \Rightarrow \phi(x) \preceq \phi(x')$.*

Whenever $(X, \preceq)$ is an extension of $(X, \preceq')$ the identity map serves as an order preserving map. An order embedding map is always order preserving, but not vice versa.

There is a rich theory on the topic of partial orders and it is still rapidly growing. Despite the simple axioms that define the structure of the poset, there is a remarkably deep theory even on finite, partially ordered sets. The number of ordered sets that can be defined on a finite set with n members, denoted with s_n, evolves as

$$\{s_n\}_1^\infty = \{1, 3, 19, 219, 4231, 130023, 6129859, 431723379, \dots\} \tag{2.1}$$

and the number of equivalence classes, i.e. classes that contain only isomorphic structures, denoted with S_n, evolves as:

$$\{S_n\}_1^\infty = \{1, 2, 5, 16, 63, 318, 2045, 16999, ...\} \tag{2.2}$$

. See Finch [34] for both of these results. This indicates how rapidly the structural variety of orders grows with increasing n. Up to now, no closed form expressions for the growth of the number of partial orders are known [34].

2.7 Cone orders

There is a large class of partial orders on $\mathbb{R}^m$ that can be defined geometrically by means of cones. In particular the so-called *cone orders* belong to this class. Cone orders satisfy two additional axioms. These are:

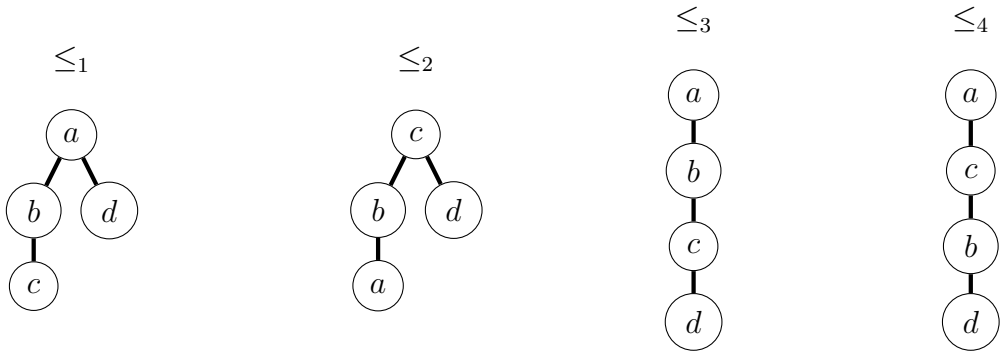

Figure 2.2: Different orders over the set $X = \{a, b, c, d\}$

Definition 22 (Translation invariance) *Let $\mathcal{R} \in \mathbb{R}^m \times \mathbb{R}^m$ denote a binary relation on $\mathbb{R}^m$. Then $\mathbb{R}$ is translation invariant, if and only if for all $\mathbf{t} \in \mathbf{R}^m$, $\mathbf{x}^1 \in \mathbb{R}^m$ and $\mathbf{x}^2 \in \mathbb{R}^m$: $\mathbf{x}^1 \mathcal{R} \mathbf{x}^2$, if and only if $(\mathbf{x}^1 + \mathbf{t})\mathcal{R}(\mathbf{x}^2 + \mathbf{t})$.*

Definition 23 (Multiplication invariance) *Let $\mathcal{R} \in \mathbb{R}^m \times \mathbb{R}^m$ denote a binary relation on $\mathbb{R}^m$. Then $\mathbb{R}$ is multiplication invariant, if and only if for all $\alpha \in \mathbf{R}$, $\mathbf{x}^1 \in \mathbb{R}^m$ and $\mathbf{x}^2 \in \mathbb{R}^m$: $\mathbf{x}^1 \mathcal{R} \mathbf{x}^2$, if and only if $(\alpha \mathbf{x}^1)\mathcal{R}(\alpha \mathbf{x}^2)$.*

We may also define these axioms on some other (vector) space on which translation and scalar multiplication is defined, but restrict ourselves to $\mathbb{R}^m$ as our interest is mainly to compare vectors of objective function values.

It has been found by V. Noghin [65] that the only partial orders on $\mathbb{R}^m$ that satisfy these two additional axioms are the cone orders on $\mathbb{R}^m$ defined by polyhedral cones. The Pareto dominance order is a special case of a strict cone order. Here the definition of strictness is inherited from the pre-order.

Cone orders can be defined geometrically and doing so provides a good intuition about their properties and minimal sets.

Definition 24 (Cone) *A subset $\mathcal{C} \subseteq \mathbb{R}^m$ is called a cone, iff $\alpha \mathbf{d} \in \mathcal{C}$ for all $\mathbf{d} \in \mathcal{C}$ and for all $\alpha \in \mathbb{R}, \alpha > 0$.*

In order to deal with cones it is useful to introduce notations for set-based calculus by Minkowski:

Definition 25 (Minkowski sum) *The Minkowski sum of two subsets S^1 and S^2 of $\mathbb{R}^m$ is defined as $S^1 + S^2 := \{s^1 + s^2 | s^1 \in S^1, s^2 \in S^2\}$. If S^1 is a singleton $\{x\}$, we may write $s + S^2$ instead of $\{s\} + S^2$.*

Definition 26 (Minkowski product) *The Minkowski product of a scalar* $\alpha \in \mathbb{R}^n$ *and a set* $S \subset \mathbb{R}^n$ *is defined as* $\alpha S := \{\alpha s | s \in S\}$.

Among the many properties that may be defined for a cone, we highlight the following two:

Definition 27 (Properties of cones) *A cone* $\mathcal{C} \in \mathbb{R}^m$ *is called:*

- *nontrivial or proper, iff* $\mathcal{C} \neq \emptyset$.
- *convex, iff* $\alpha \mathbf{d}^1 + (1-\alpha)\mathbf{d}^2 \in \mathcal{C}$ *for all* $\mathbf{d}^1$ *and* $\mathbf{d}^2 \in \mathcal{C}$ *for all* $0 < \alpha < 1$
- *pointed, iff for* $\mathbf{d} \in \mathcal{C}, \mathbf{d} \neq 0, -\mathbf{d} \notin \mathcal{C}$, *i e.* $\mathcal{C} \cap -\mathcal{C} \subseteq \{0\}$

Example As an example of a cone consider the possible futures of a particle in a 2-D world that can move with a maximal speed of c in all directions: This cone is defined as $\mathcal{C}^+ = \{\mathcal{D}(t) | t \in \mathbb{R}^+\}$, where $\mathcal{D}(t) = \{\mathbf{x} \in \mathbb{R}^3 | (x_1)^2 + (x_2)^2 \leq (ct)^2, x_3 = t\}$. Here time is measured by negative and positive values of t, where $t = 0$ represents the current time. We may ask now, whether given the current position $\mathbf{x}_0$ of a particle, a locus $\mathbf{x} \in \mathbb{R}^3$ is a possible future of the particle. The answer is in the affirmative, iff $\mathbf{x}_0$ if $\mathbf{x} \in \mathbf{x}_0 + \mathcal{C}^+$.

We will now can define Pareto dominance and the weak (strict) componentwise order by means of dominance cones. For this we have to define special convex cones in $\mathbb{R}$:

Definition 28 (Orthants) *We define*

- *the* positive orthant $\mathbb{R}^n_{\geq} := \{\mathbf{x} \in \mathbb{R}^n | x_1 \geq 0, \ldots, x_n \geq 0\}$.
- *the* null-dominated orthant $\mathbb{R}^n_{\prec_{pareto}} := \{\mathbf{x} \in \mathbb{R}^n | 0 \prec_{pareto} \mathbf{x}\}$.
- *the* strictly positive orthant $\mathbb{R}^n_{>} := \{\mathbf{x} \in \mathbb{R}^n | x_1 > 0, \ldots, x_n > 0\}$.

Now, let us introduce the alternative definitions for Pareto dominance:

Definition 29 (Pareto dominance) *Given two vectors* $\mathbf{x} \in \mathbb{R}^\mathbf{n}$ *and* $\mathbf{x}' \in \mathbb{R}^\mathbf{n}$*:*

- $\mathbf{x} < \mathbf{x}'$ *(in symbols:* $\mathbf{x}$ *dominates* $\mathbf{x}'$*) in the strict componentwise order* $\Leftrightarrow$ $\mathbf{x}' \in \mathbf{x} + \mathbb{R}^n_{>}$
- $\mathbf{x} \prec \mathbf{x}'$ *(in symbols:* $\mathbf{x}$ *dominates* $\mathbf{x}'$*)* $\Leftrightarrow \mathbf{x}' \in \mathbf{x} + \mathbb{R}^n_{\prec_{pareto}}$
- $\mathbf{x} \geq \mathbf{x}'$ *(in symbols:* $\mathbf{x}$ *dominates* $\mathbf{x}'$*) in the weak componentwise order* $\Leftrightarrow$ $\mathbf{x}' \in \mathbf{x} - \mathbb{R}^n_{\geq}$

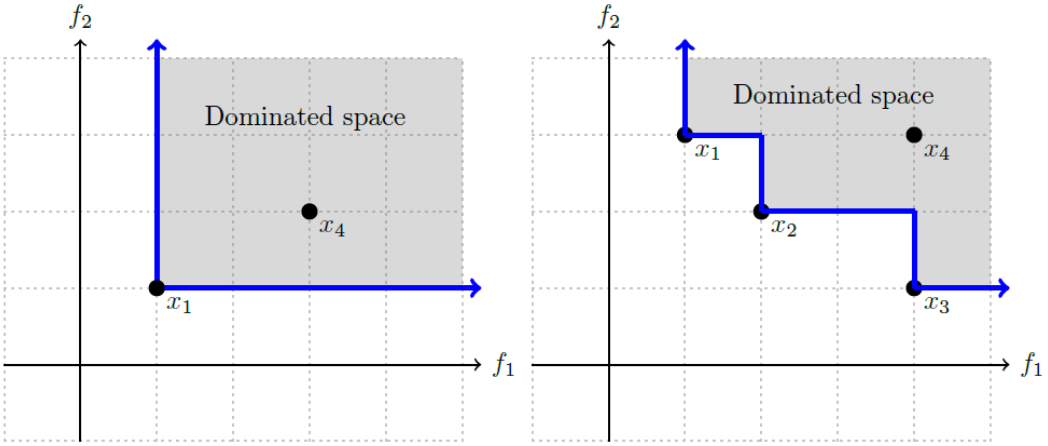

Figure 2.3: Pareto domination in $\mathbb{R}^2$ defined by means of cones. In the left hand side of the figure the points inside the dominated region are dominated by $\mathbf{x}$. In the figure on the right side the set of points dominated by the set $A = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$ is depicted.

It is often easier to assess graphically whether a point dominates another point by looking at cones (cf. Fig. 2.3 (l)). This holds also for a region that is dominated by a *set of points*, such that at least one point from the set dominates it (cf. Fig. 2.3 (r)).

Definition 30 (Dominance by a set of points) *A point* $\mathbf{x}$ *is said to be dominated by a set of points* A *(notation:* $A \prec \mathbf{x}$*, iff* $\mathbf{x} \in A + \mathbb{R}^n_{\prec}$*, i. e. iff there exists a point* $\mathbf{x}' \in \mathbf{A}$*, such that* $\mathbf{x}' \prec_{Pareto} \mathbf{x}$*.*

In the theory of multiobjective and constrained optimization, so-called polyhedral cones play a crucial role.

Definition 31 *A cone* $\mathcal{C}$ *is a polyhedral cone with a finite basis, if and only if there is a set of vectors* $D = \{\mathbf{d}_1, \ldots, \mathbf{d}_k\} \subset \mathbb{R}^m$ *and* $\mathcal{C} = \{\lambda_1\mathbf{d}_1 + \cdots + \lambda_k\mathbf{d}_k | \lambda \in \mathbb{R}^+_0, i = 1, \ldots, k\}$.

By chosing the coordinate vectors $\mathbf{e}_i$ to construct a polyhedral cones that resemble of the weak componentwise order.

Example In figure 2.4 an example of an cone is depicted with finite basis $D = \{\mathbf{d}_1, \mathbf{d}_2\}$ and $\mathbf{d}_1 = (2, 1)^\top, \mathbf{d}_2 = (1, 2)^\top$. It is defined as

$$C := \{\lambda_1\mathbf{d}_1 + \lambda_2\mathbf{d}_2 |\, \lambda_1 \in [0, \infty], \lambda_2 \in [0, \infty]\}\,.$$

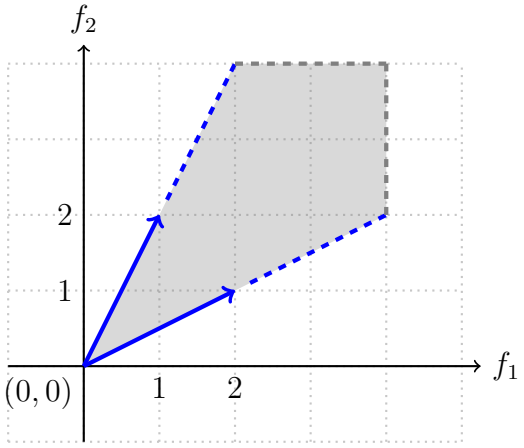

Figure 2.4: Dominance cone for cone order in Example 2.7.

This cone is pointed, because $C \cap -C = \emptyset$. Moreover, C is a convex cone. This is because two points in C, say $\mathbf{p}_1$ and $\mathbf{p}_2$ can be expressed by $\mathbf{p}_1 = \lambda_{11}\mathbf{d}_1 + \lambda_{21}\mathbf{d}_2$ and $\mathbf{p}_2 = \lambda_{12}\mathbf{d}_1 + \lambda_{22}\mathbf{d}_2$, $\lambda_{ij} \in [0, \infty)$, $i = 1, 2; j = 1, 2$. Now, for a given $\lambda \in [0, 1]$ $\lambda\mathbf{p}_1 + (1-\lambda)\mathbf{p}_2$ equals $\lambda\lambda_{11}\mathbf{d}_1 + (1-\lambda)\lambda_{12}\mathbf{d}_1 + \lambda\lambda_{21}\mathbf{d}_2 + (1-\lambda)\lambda_{22}\mathbf{d}_2 =: c_1\mathbf{d}_1 + c_2\mathbf{d}_2$, where it holds that $c_1 \in [0, \infty)$ and $c_2 \in [0, \infty)$. According to the definition of C the cone therefore the point $\lambda\mathbf{p}_1 + (1 - \lambda)\mathbf{p}_2$ is part of the cone C.

Further topics related to cone orders are addressed in [28].

Exercises

1. In definition 6 some common properties are defined that binary relations can have and some examples are given below. Find further examples from real-life for binary relations! Which axioms from definition 6 do they obey!

2. Characterize incomparability (definition 10) axiomatically! What are the essential differences to indifference?

3. Describe the Pareto order on the set of 3-D hypercube edges $\{(0, 1, 0)^T$, $(0, 0, 1)^T, (1, 0, 0)^T, (0, 0, 0)^T, (0, 1, 1)^T, (1, 0, 1), (1, 1, 0)^T, (1, 1, 1)^T\}$ by means of the graph of a binary relation and by means of the Hasse diagram!

4. Prove, that $(\mathbb{N} - \{1\}, \preceq)$ with $a \preceq b \Leftrightarrow a \mod b \equiv 0$ is a partially ordered set. What are the minimal (maximal) elements of this set?

5. Prove that the time cone $\mathcal{C}^+$ is convex! Compare the Pareto order to the order defined by time cones!

Chapter 3

Landscape Analysis

In this chapter we will come back to optimization problems, as defined in the first chapter. We will introduce different notions of Pareto optimality and discuss necessary and sufficient conditions for (Pareto) optimality and efficiency in the constrained and unconstrained case. In many cases, optimality conditions directly point to solution methods for optimization problems. As in Pareto optimization there is rather a set of optimal solutions then a single optimal solution, we will also look at possible structures of optimal sets.

3.1 Search Space vs. Objective Space

In Pareto optimization we are considering two spaces - the *decision space* or *search space* $\mathbb{S}$ and the *objective space* $\mathbb{Y}$. The vector valued objective function $\mathbf{f} : \mathbb{S} \to \mathbb{Y}$ provides the mapping from the decision space to the objective space. The set of feasible solutions $\mathcal{X}$ can be considered as a subset of the decision space, i. e. $\mathcal{X} \subseteq \mathbb{S}$. Given a set $\mathcal{X}$ of feasible solutions, we can define $\mathcal{Y}$ as the image of $\mathcal{X}$ under $\mathbf{f}$.

The sets $\mathbb{S}$ and $\mathbb{Y}$ are usually not arbitrary sets. If we want to define optimization tasks, it is mandatory that an order structure is defined on $\mathbb{Y}$. The space $\mathbb{S}$ is usually equipped with a neighborhood structure. This neighborhood structure is not needed for defining global optima, but it is exploited, however, by optimization algorithms that gradually approach optima and in the formulation of local optimality conditions. Note, that the choice of neighborhood system may influence the difficulty of an optimization problem significantly. Moreover, we note that the definition of neighborhood gives rise to many characterizations of functions, such as local optimality and barriers. Especially in discrete spaces the neighborhood structure needs to be mentioned then, while in continuous optimization locality mostly refers to the Euclidean metric.

The definition of landscape is useful to distinguish the general concept of a

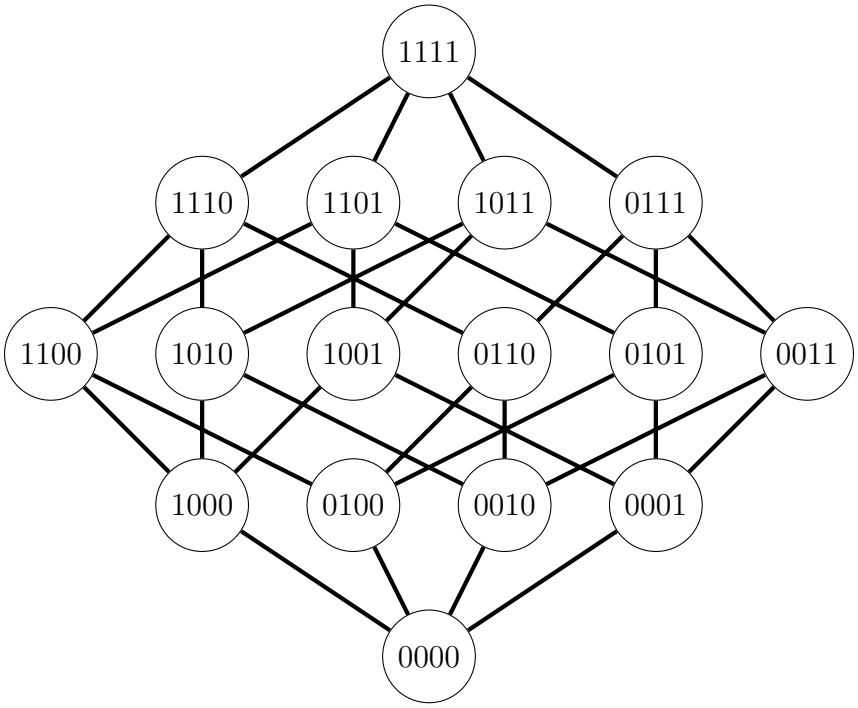

Figure 3.1: The 'binland' landscape of the bitstring $\{0,1\}^4$, with edges representing a Hamming distance of 1, is an example of a discrete, partially ordered landscape.

function from the concept of a function with a neighborhood defined on the search space and a (partial) order defined on the objective space. We define (poset valued) landscapes as follows:

Hasse diagram of the Pareto order for the leading ones trailing zeros (LOTZ) problem. The first objective is to maximize the number of leading ones in the bitstring, while the second objective is to maximize the number of trailing zeros. The preorder on $\{0,1\}$ is then defined by the Pareto dominance relation. In this example all local minima are also global minima.

Definition 32 *A poset valued landscape is a quadruple* $\mathcal{L} = (\mathcal{X}, N, \mathbf{f}, \preceq)$ *with* $\mathcal{X}$ *being a set and* N *a neighborhood system defined on it (e.g. a metric).* $\mathbf{f} : \mathcal{X} \to \mathbb{R}^m$ *is a vector function and* $\preceq$ *a partial order defined on* $\mathbb{R}^m$*. The function* $\mathbf{f} : \mathcal{X} \to \mathbb{R}^m$ *will be called* height function*.*

An example for a poset-valued landscape is given in the Figure 3.9 and Figure 3.2. Here the neighborhood system is defined by the Hamming distance. It gets obvious that in order to define a landscape in finite spaces we need two essential structures. A neighborhood graph in search space (where edges connect nearest neighbors) the Hasse diagram on the objective space.

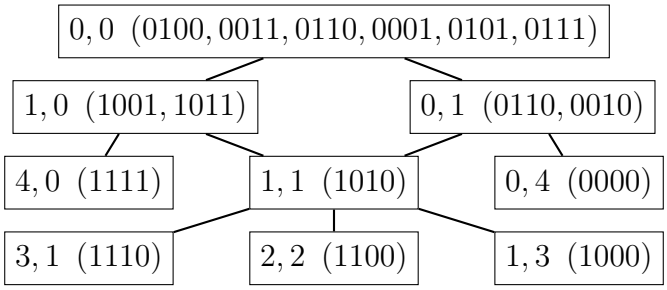

Figure 3.2: (Figure 3.2) Hasse diagram of the Pareto order for the leading ones trailing zeros (LOTZ) problem. The first objective is to maximize the number of leading ones in the bitstring, while the second objective is to maximize the number of trailing zeros. The preorder on $\{0,4\}^2$ is then defined by the Pareto dominance relation. In this example all local minima are also global minima (cf. Fig. 3.9).

Note, that for many definitions related to optimization we do not have to specify a height function and it suffices to define an order on the search space. For concepts like global minima the neighborhood system is not relevant either. Therefore, this definition should be understood as a kind of superset of the structure we may refer to in multicriteria optimization.

3.2 Global Pareto Fronts and Efficient Sets

Given $\mathbf{f} : \mathbb{S} \to \mathbb{R}^m$. Here we write $\mathbf{f}$ instead of $(f_1, \ldots, f_m)^\top$. Consider an optimization problem:

$$\mathbf{f}(\mathbf{x}) \to \min, \mathbf{x} \in \mathcal{X} \tag{3.1}$$

Recall that the Pareto front and the efficient set are defined as follows (Section 1.5):

Definition 33 *Pareto front and efficient set*
The Pareto front $\mathcal{Y}_N$ is defined as the set of non-dominated solutions in $\mathcal{Y} = \mathbf{f}(\mathcal{X})$, i. e. $\mathcal{Y}_N = \{\mathbf{y} \in \mathcal{Y} \mid \nexists \mathbf{y}' \in \mathcal{Y} : \mathbf{y}' \prec \mathbf{y}\}$. The efficient set is defined as the pre-image of the Pareto-front, $\mathcal{X}_E = f^{-1}(\mathcal{Y}_N)$.

Note, that the cardinality $\mathcal{X}_E$ is at least as big as $\mathcal{Y}_N$, but not vice versa, because there can be more than one point in $\mathcal{X}_E$ with the same image in $\mathcal{Y}_N$. The elements of $\mathcal{X}_E$ are termed efficient points.

In some cases it is more convenient to look at a direct definition of efficient points:

Definition 34 *A point $\mathbf{x}^{(1)} \in \mathcal{X}$ is* efficient*, iff $\nexists \mathbf{x}^{(2)} \in \mathcal{X} : \mathbf{x}^{(2)} \prec \mathbf{x}^{(1)}$.*

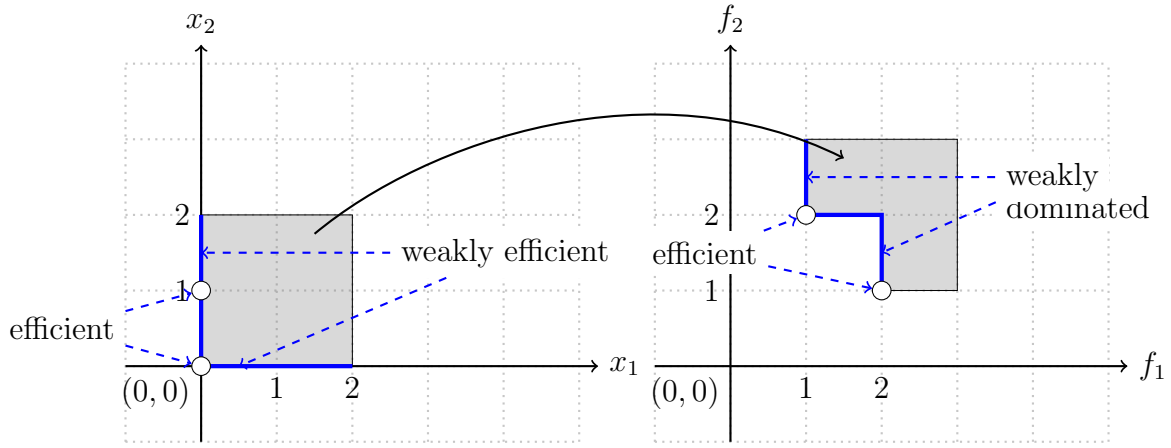

Figure 3.3: Example of a solution set containing efficient solutions (open points) and weakly efficient solutions (thick blue line).

Again, the set of all efficient solutions in $\mathcal{X}$ is denoted as $\mathcal{X}_E$.

Remark Efficiency is always relative to a set of solutions. In future, we will not always consider this set to be the entire search space of an optimization problem, but we will also consider the efficient set of a subset of the search space. For example the efficient set for a finite sample of solutions from the search space that has been produced so far by an algorithm may be considered as a temporary approximation to the efficient set of the entire search space.

3.3 Weak efficiency

Besides the concept of *efficiency* also the concept of *weak efficiency*, for technical reasons, is important in the field of multicriteria optimization. For example points on the boundary of the dominated subspace are often characterized as weakly efficient solutions though they may be not efficient.

Recall the definition of strict domination (Section 1.5):

Definition 35 *Strict dominance*

Let $\mathbf{y}^{(1)}, \mathbf{y}^{(2)} \in \mathbb{R}^m$ *denote two vectors in the objective space. Then* $\mathbf{y}^{(1)}$ *strictly dominates* $\mathbf{y}^{(2)}$ *(in symbols:* $\mathbf{y}^{(1)} < \mathbf{y}^{(2)}$*), iff* $\forall i = 1, \ldots, m : y_i^{(1)} < y_i^{(2)}$.

Definition 36 *Weakly efficient solution*

A solution $\mathbf{x}^{(1)} \in \mathcal{X}$ *is weakly efficient, iff* $\nexists \mathbf{x}^{(2)} \in \mathcal{X} : \mathbf{f}(\mathbf{x}^{(2)}) < \mathbf{f}(\mathbf{x}^{(1)})$. *The set of all weakly efficient solutions in* $\mathcal{X}$ *is called* $\mathcal{X}_{wE}$.

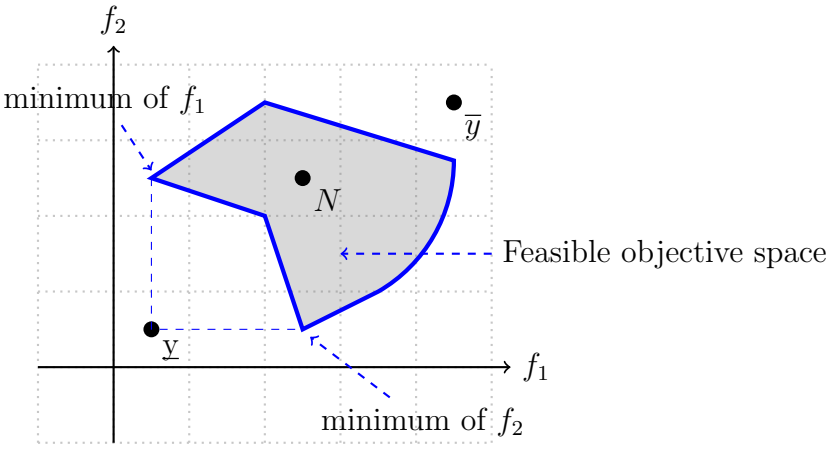

Figure 3.4: The shaded region indicates the feasible objective space of some function. Its *ideal point*, $\underline{y}$, its *Nadir point*, N and its *maximal point*, $\overline{y}$, are visible.

Example In Fig. 3.3 we graphically represent the efficient and weakly efficient set of the following problem: $f = (f_1, f_2) \to \min, \mathbb{S} = \mathcal{X} = [0,2] \times [0,2]$, where f_1 and f_2 are as follows:

$$f_1(x_1, x_2) = \begin{cases} 2 + x_1 & \text{if } 0 \leq x_2 < 1 \\ 1 + 0.5x_1 & \text{otherwise} \end{cases}, f_2(x_1, x_2) = 1 + x_1, x_1 \in [0,2], x_2 \in [0,2].$$

. The solutions $(x_1, x_2) = (0,0)$ and $(x_1, x_2) = (0,1)$ are efficient solutions of this problem, while the solutions on the line segments indicated by the bold line segments in the figure denote weakly efficient solutions. Note, that both efficient solutions are also weakly efficient, as efficiency implies weak efficiency.

3.4 Characteristics of Pareto Sets

There are some characteristic points on a Pareto front:

Definition 37 *Given an multi-objective optimization problem with m objective functions and image set $\mathcal{Y}$: The ideal solution is defined as*

$$\underline{\mathbf{y}} = (\min_{\mathbf{y} \in \mathcal{Y}} y_1, \ldots, \min_{\mathbf{y} \in \mathcal{Y}} y_m).$$

Accordingly we define the maximal solution:

$$\overline{\mathbf{y}} = (\max_{\mathbf{y} \in \mathcal{Y}} y_1, \ldots, \max_{\mathbf{y} \in \mathcal{Y}} y_m).$$

The Nadir point is defined:

$$\mathbf{y}^N = (\max_{\mathbf{y}\in\mathcal{Y}_N} y_1, \ldots, \max_{\mathbf{y}\in\mathcal{Y}_N} y_m).$$

For the Nadir only points from the Pareto front $\mathcal{Y}_N$ are considered, while for the maximal point all points in $\mathcal{Y}$ are considered. The latter property makes it, for dimensions higher than two ($m > 2$), more difficult to compute the Nadir point. In that case the computation of the Nadir point cannot be reduced to m single criterion optimizations.

A visualization of these entities in a 2-D space is given in figure 3.4.

3.5 Optimality conditions based on level sets

Level sets can be used to visualize $\mathcal{X}_E$, $\mathcal{X}_{wE}$ and $\mathcal{X}_{sE}$ for continuous spaces and obtain these sets graphically in the low-dimensional case: Let in the following definitions f be a function $f : \mathbb{S} \to \mathbb{R}$, for instance one of the objective functions:

Definition 38 *Level sets*

$$\mathcal{L}_{\leq}(f(\hat{\mathbf{x}})) = \{\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) \leq f(\hat{\mathbf{x}})\} \tag{3.2}$$

Definition 39 *Level curves*

$$\mathcal{L}_{=}(f(\hat{\mathbf{x}})) = \{\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) = f(\hat{\mathbf{x}})\} \tag{3.3}$$

Definition 40 *Strict level set*

$$\mathcal{L}_{<}(f(\hat{\mathbf{x}})) = \{\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) < f(\hat{\mathbf{x}})\} \tag{3.4}$$

Level sets can be used to determine whether $\hat{\mathbf{x}} \in \mathcal{X}$ is (strictly, weakly) non-dominated or not.

The point $\hat{\mathbf{x}}$ can only be efficient if its level sets intersect in level curves.

Theorem 41 $\mathbf{x}$ *is efficient* $\Leftrightarrow \bigcap_{k=1}^{m} \mathcal{L}_{\leq}(f_k(\mathbf{x})) = \bigcap_{k=1}^{m} \mathcal{L}_{=}(f_k(\mathbf{x}))$

Proof: $\hat{\mathbf{x}}$ is efficient $\Leftrightarrow$ there is no $\mathbf{x}$ such that both $f_k(\mathbf{x}) \leq f_k(\hat{\mathbf{x}})$ for all $k = 1, \ldots, m$ and $f_k(\mathbf{x}) < f(\hat{\mathbf{x}})$ for at least one $k = 1, \ldots, m \Leftrightarrow$ there is no $\mathbf{x} \in \mathcal{X}$ such that both $\mathbf{x} \in \cap_{k=1}^{m} \mathcal{L}_{\leq}(f(\hat{\mathbf{x}}))$ and $\mathbf{x} \in \mathcal{L}_{<}(f_j(\hat{\mathbf{x}}))$ for some $j \Leftrightarrow \bigcap_{k=1}^{m} \mathcal{L}_{\leq}(f_k(\hat{x})) = \bigcap_{k=1}^{m} \mathcal{L}_{=}(f_k(\hat{x}))$

Theorem 42 *The point* $\hat{\mathbf{x}}$ *can only be weakly efficient if its strict level sets do not intersect.* $\mathbf{x}$ *is weakly efficient* $\Leftrightarrow \bigcap_{k=1}^{m} \mathcal{L}_{<}(f_k(\mathbf{x})) = \emptyset$

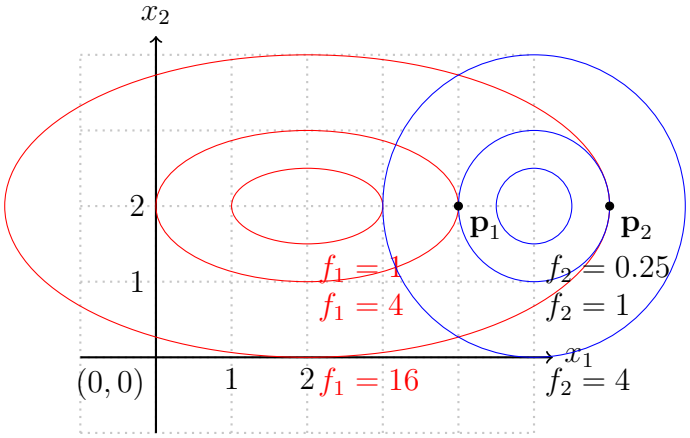

Figure 3.5: This graph depicts the level curves of $f_1(\mathbf{x}) = (x_1-2)+2(x_2-2) \to \min$ (red curves) and $f_2(\mathbf{x}) = (x_1-5)+(x_2-2) \to \min$ (blue curves).

Theorem 43 *The point $\hat{\mathbf{x}}$ can only be strictly efficient if its level sets intersect in exactly one point. $\mathbf{x}$ is strictly efficient* $\Leftrightarrow \bigcap_{k=1}^{m} \mathcal{L}_{\leq}(f_k(\mathbf{x})) = \{\mathbf{x}\}$

Level sets have a graphical interpretation that helps to geometrically understand optimality conditions and landscape characteristics. Though this intuitive geometrical interpretation may only be viable for lower dimensional spaces, it can help to develop intuition about problems in higher dimensional spaces. The visualization of level sets can be combined with the visualization of constraints, by partitioning the search space into a feasible and infeasible part.

The following examples will illustrate the use of level sets for visualization:

Example Consider the problem $f_1(x_1, x_2) = (x_1 - 1.75)^2 + 4(x_2 - 1)^2 \to \min$, $f_2(x_1, x_2) = (x_1 - 3)^2 + (x_2 - 1)^2 \to \min, (x_1, x_2)^\top \in \mathbb{R}^2$. The level curves of this problem are depicted in Figure 3.5 together with the two marked points $\mathbf{p}_1$ and $\mathbf{p}_2$ that we will study now. For $\mathbf{p}_1$ it gets clear from Figure 3.6 that it is an efficient point as it cannot be improved in both objective function values at the same time. On the other hand $\mathbf{p}_2$ is no level point as by moving it to the region directly left of it can be improved in all objective function values at the same time. Formally, the existence of such a region follows from the non-empty intersection of $\mathcal{L}_<(f_1(\mathbf{p}_2))$ and $\mathcal{L}_<(f_2(\mathbf{p}_2))$.

Example Consider the search space $\mathcal{S} = [0, 2] \times [0, 3]$. Two objectives $f_1(x_1, x_2) = 2 + \frac{1}{3}x_2 - x_1 \to \min, \quad f_2(x_1, x_2) = \frac{1}{2}x_2 + \frac{1}{2}x_1 \to \max$, In addition the constraint $g(x_1, x_2) = -\frac{2}{3}x_1 - x_2 \geq 0$ needs to be fulfilled. To solve this problem, we mark the constrained region graphically (see Figure 3.7) Now, we can check different

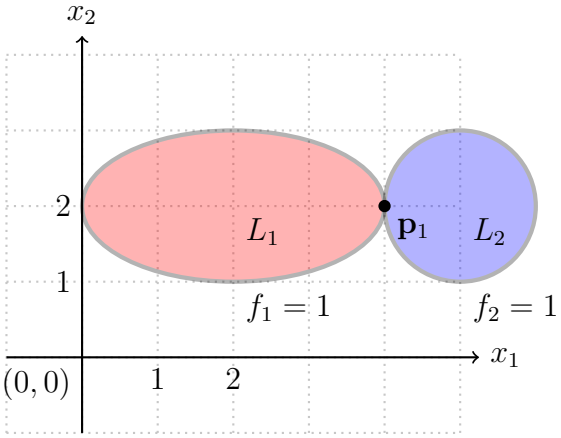

Figure 3.6: The situation for $\mathbf{p}_1$: In order to improve f_1 the point $\mathbf{p_1}$ has to move into the set $\mathcal{L}_{\leq}(f_1(\mathbf{p}_1))$ and in order to improve f_2 it needs to move into $\mathcal{L}_{\leq}(f_1(\mathbf{p}_1))$. Since these sets only meet in $\mathbf{p}_1$, it is not possible to improve f_1 and f_2 at the same time.

points for efficiency. For $\mathbf{p}_1$ the region where both objectives improve is in the upper triangle bounded by the level curves. As this set is partly feasible, it is possible to find a dominating feasible point and $\mathbf{p}_1$ is not efficient. In contrast, for $\mathbf{p}_2$ the set of dominating solutions is completely in the infeasible domain, why this point belongs to the efficient set. The complete efficient set in this example lies on the constraint boundary. Generally, it can be found that for linear problems with level curves intersecting in a single point there exists no efficient solutions in the unconstrained case whereas efficient solutions may lie on the constraint boundary in the constrained case.

3.6 Local Pareto Optimality

As opposed to global Pareto optimality we may also define local Pareto optimality. Roughly speaking, a solution is a local optimum, if there is no better solution in its neighborhood. In order to put things into more concrete terms let us distinguish continuous and discrete search spaces:

In finite discrete search spaces for each point in the search space $\mathbb{X}$ a set of nearest neighbors can be defined by means of some *n*eighborhood function $N : \mathbb{X} \to \wp(\mathbb{X})$ with $\forall x \in \mathbb{X} : x \notin N(x)$. As an example consider the space $\{0, 1\}^n$ of bit-strings of length n with the nearest neigbors of a bit-string x being the elements that differ only in a single bit, i.e. that have a Hamming distance of 1.

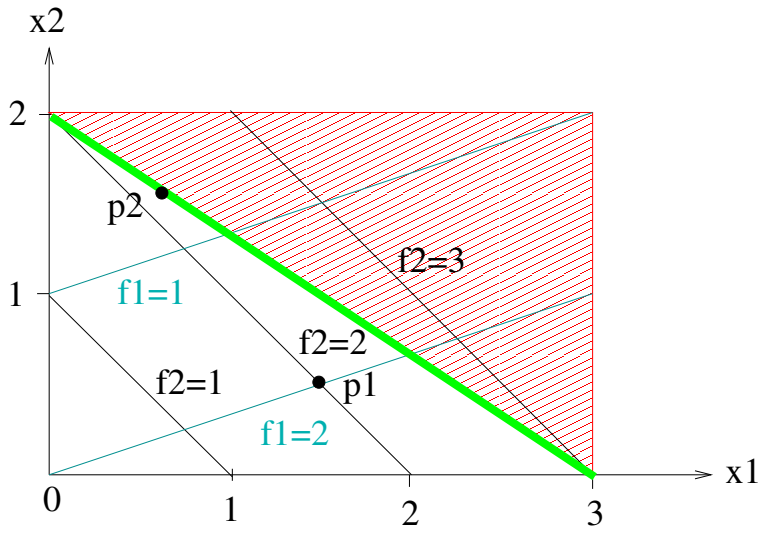

Figure 3.7: Example for a linear programming problem visualization with two objective functions

Definition 44 *Locally efficient point (finite search spaces)*
Given a neighborhood function $N : \mathbb{X} \to \wp(\mathbb{X})$, a locally efficient solution is a point $x \in \mathbb{X}$ such that $\nexists x' \in N(x) : x' \prec x$.

Definition 45 *Strictly locally efficient point (finite search spaces)*
Given a neighborhood function $N : \mathbb{X} \to \wp(\mathbb{X})$, a strictly locally efficient solution is a point $x \in \mathbb{X}$ such that $\nexists x' \in N(x) : x' \preceq x$.

Remark: The comparison of two elements in the search space is done in the objective space. Therefore, for two elements x and x' with $x \preceq x'$ and $x \preceq x'$ it can happen that $x \neq x'$ (see also the discussion of the antisymmetry property in chapter 2).

This definition can also be extended for countable infinite sets, though we must be cautious with the definition of the neighborhood function there.

For the Euclidean space $\mathbb{R}^n$, the notion of nearest neighbors does not make sense, as for every point different from some point x there exists another point different from x that is closer to x. Here, the following criterion can be used to classify local optima:

Definition 46 *Open ϵ-ball*
An open ϵ-ball $B_\epsilon(x)$ around a point $x \in \mathbb{R}^n$ is defined as: $B_\epsilon(x) = \{x' \in \mathbb{X} | d(x, x') < \epsilon\}$.

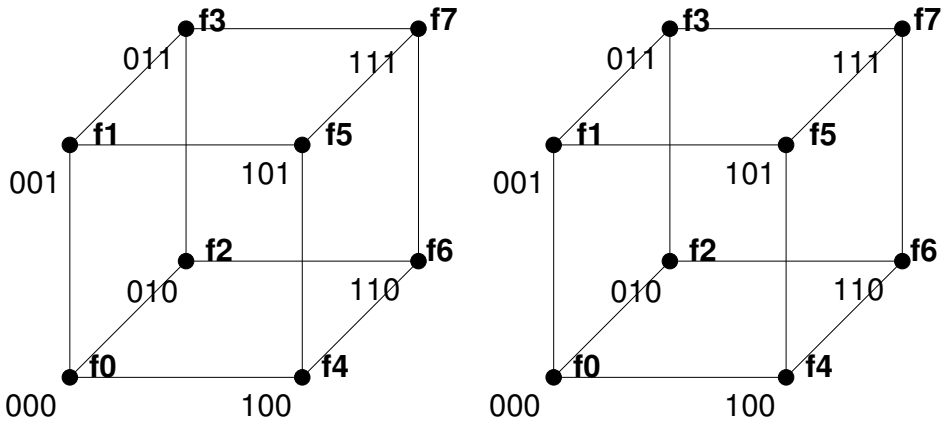

Figure 3.8: Pseudoboolean landscapes with search space $\{0,1\}^3$ and the Hamming neighborhood defined on it. The linearly ordered landscape on the right hand side has three local optima. These are $x^{(0)} = 000, x^{(4)} = 100, x^{(7)} = 111$. $x^{(0)}$ is also a global minimum and $x^{(4)}$ a global maximum. The partially ordered landscape on the right hand side has locally efficient solutions are $x^{(1)} = 001, x^{(2)} = 010, x^{(3)} = 011, x^{(6)} = 110$. The globally efficient solutions are $x^{(1)}, x^{(2)}$ and $x^{(3)}$

Definition 47 *Locally efficient point (Euclidean search spaces)*
A point $x \in \mathbb{R}^n$ in a metric space is said to be a locally efficient solution, iff $\exists \epsilon > 0 : \nexists x' \in B_\epsilon(x) : x' \prec x$.

Definition 48 *Strictly locally efficient point (Euclidean search spaces)*
A point $x \in \mathbb{R}^n$ in a metric space is said to be a strictly locally efficient solution, iff $\exists \epsilon > 0 : \nexists x' \in B_\epsilon(x) - \{x\} : x' \preceq x$.

The extension of the concept of local optimality can be done also for subspaces of the Euclidean space, such as box constrained spaces as in definition 1.8.

3.7 Barrier Structures

Local optima are just one of the many characteristics we may discuss for landscapes, i.e. functions with a neighborhood structure defined on the search space. Looking at different local optimal of a landscape we may ask ourselves how these local optimal are separated from each other. Surely there is some kind of barrier in between, i.e. in order to reach one local optimum from the other following a path of neighbors in the search space we need to put up with encountering worsening of solutions along the path. We will next develop a formal framework on defining barriers and their characteristics and highlight an interesting hierarchical structure

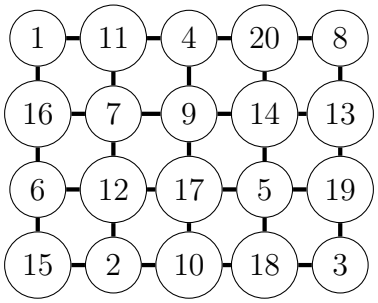

Figure 3.9: Example of a discrete landscape. The height of points is given by the numbers and their neighborhood is expressed by the edges.

that can be obtained for all landscapes - the so called barrier tree of totally ordered landscapes, which generalizes to a barrier forest in partially ordered landscapes.

For the sake of clarity let us introduce formal definitions first for landscapes with a one-dimensional height function as they are discussed in single-objective optimization.

Definition 49 *Path in discrete spaces*

Let $N : \mathbb{X} \to \wp(\mathbb{X})$ *be a neighborhood function. A sequence* $\mathbf{p}_1, \ldots, \mathbf{p}_l$ *for some* $l \in \mathbb{N}$ *and* $\mathbf{p}_1, \ldots, \mathbf{p}_l \in \mathbb{S}$ *is called a* path connecting $\mathbf{x}_1$ and $\mathbf{x}_2$, iff $\mathbf{p}_1 = \mathbf{x}_1$, $\mathbf{p}_{i+1} \in N(\mathbf{p}_i)$, for $i = 1, \ldots, l-1$, and $\mathbf{p}_l = \mathbf{x}_2$.

Definition 50 *Path in continuous spaces*

For continuous spaces a path is a continuous mapping $\mathbf{p}[0,1] \to \mathbb{X}$ *with* $\mathbf{p}(0) = \mathbf{x}_1$ *and* $\mathbf{p}(1) = \mathbf{x}_2$.

Definition 51 *Let* $\mathbb{P}_{\mathbf{x}_1,\mathbf{x}_2}$ *denote the set of all paths between* $\mathbf{x}_1$ *and* $\mathbf{x}_2$.

Definition 52 *Let the function value of the* lowest point *separating two local minima* $\mathbf{x}_1$ *and* $\mathbf{x}_2$ *be defined as* $\hat{f}(\mathbf{x}_1, \mathbf{x}_2) = \min_{\mathbf{p} \in \mathbb{P}_{\mathbf{x}_1,\mathbf{x}_2}} \max_{\mathbf{x}_3 \in \mathbf{p}} f(\mathbf{x}_3)$. *Points* $\mathbf{s}$ *on some path* $\mathbf{p} \in \mathbb{P}_{\mathbf{x}_1,\mathbf{x}_2}$ *for which* $f(\mathbf{s}) = \hat{f}(\mathbf{x}_1, \mathbf{x}_2)$ *are called saddle points between* $\mathbf{x}_1$ *and* $\mathbf{x}_2$.

Example In the example given in Figure 3.9 the search points are labeled by their heights, i.e. x_1 has height 1 and x_4 has height 4. The saddle point between the local minima x_1 and x_2 is x_{12}. The saddle point x_3 and x_5 is x_{18}.

Lemma 53 *For* non-degenerate landscapes*, i.e. landscapes where for all* $\mathbf{x}_1$ *and* $\mathbf{x}_2$*:* $f(\mathbf{x}_1) \neq f(\mathbf{x}_2)$*, saddle points between two given local optima are unique.*

Note. that in case of degenerate landscapes, i.e. landscapes where there are at least two different points which share the same value of the height function, saddle points between two given local optima are not necessarily unique anymore, which, as we will see later, influences the uniqueness of barrier trees characterizing the overall landscape.

Definition 54 *The valley (or: basin) below a point* $\mathbf{s}$ *is called* $B(\mathbf{s}) : B(\mathbf{s}) = \{\mathbf{x} \in \mathbb{S} | \exists \mathbf{p} \in \mathbb{P}_{\mathbf{x},\mathbf{s}} : \max_{z \in \mathbf{p}} f(\mathbf{z}) \leq f(\mathbf{s})\}$

Example In the aforementioned example given in Figure 3.9, Again, search points $x_1, \ldots, x_{20}$ are labeled by their heights, i.e. x_4 is the point with height 4, etc.. The basin below x_1 is given by the empty set, and the basin below x_{14} is $\{x_1, x_{11}, x_4, x_9, x_7, x_{13}, x_5, x_8, x_{14}, x_{12}, x_2, x_6, x_{10}\}$.

Points in $B(\mathbf{s})$ are mutually connected by paths that never exceed $f(\mathbf{s})$. At this point it is interesting to compare the level set $\mathcal{L}_{\leq}(f(\mathbf{x}))$ with the basin $B(\mathbf{x})$. The connection between both concepts is: Let $\mathcal{B}$ be the set of connected components of the level set $\mathcal{L}_{\leq}(f(\mathbf{x}))$ with regard to the neighborhood graph of the search space $\mathcal{X}$, then $B(\mathbf{x})$ is the connected component in which $\mathbf{x}$ resides.

Theorem 55 *Suppose for two points* $\mathbf{x}_1$ *and* $\mathbf{x}_2$ *that* $f(\mathbf{x}_1) \leq f(\mathbf{x}_2)$*. Then, either* $B(\mathbf{x_1}) \subseteq B(\mathbf{x}_2)$ *or* $B(\mathbf{x}_1) \cap B(\mathbf{x}_2) = \emptyset$. $\square$

Theorem 55 implies that the barrier structure of a landscapes can be represented as a tree where the saddle points are the branching points and the local optima are the leaves. The *flooding algorithm* (see Algorithm 2) can be used for the construction of the barrier tree in discrete landscapes with finite search space $\mathcal{X}$ and linearly ordered search points (e.g. by means of the objective function values). Note, that if the height function is not injective the flooding algorithm can still be used but the barrier tree may not be uniquely defined. The reason for this is that there are different possibilities of how to sort elements with equal heights in line 1 of algorithm 2.

Finally, let us look whether concepts such as saddle points, basins, and barrier trees can be generalized in a meaningful way for partially ordered landscapes. Flamm and Stadler [76] recently proposed one way of generalizing these concepts. We will review their approach briefly and refer to the paper for details.

Adjacent points in linearly ordered landscapes are always comparable. This does not hold in general for partially ordered landscapes. We have to modify the paths $\mathbf{p}$ that enter the definition.

Definition 56 *Maximal points on a path*
The set of maximal points on a path $\mathbf{p}$ *is defined as* $\sigma(\mathbf{p}) = \{\mathbf{x} \in \mathbf{p} | \nexists \mathbf{x}' \in \mathbf{p} : f(\mathbf{x}) \prec f(\mathbf{x}')\}$

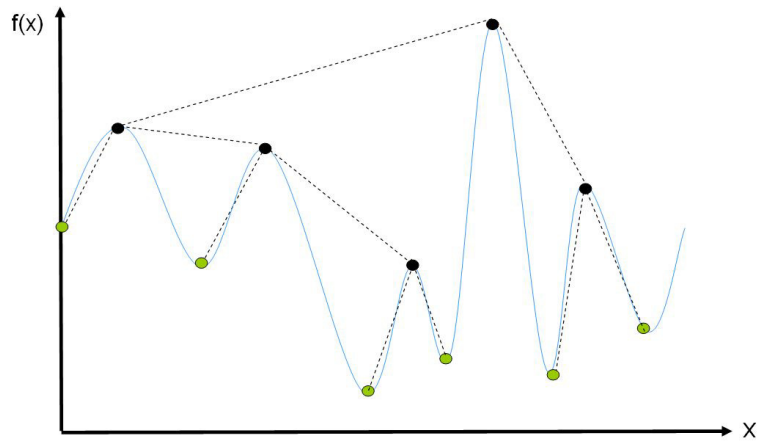

Figure 3.10: A barrier tree of a 1-D continuous function.

Definition 57 *Poset saddle-points*
$\Sigma_{\mathbf{x}_1,\mathbf{x}_2} = \bigcup_{\mathbf{p}\in\mathbb{P}_{\mathbf{x}_1,\mathbf{x}_2}} \sigma(\mathbf{p})$ *is the set of maximal elements along all possible paths.* Poset-saddle points *are defined as the Pareto optima*[1] *of* $\Sigma_{\mathbf{x}_1,\mathbf{x}_2}$*:* $S(\mathbf{x}_1,\mathbf{x}_2) := \{\mathbf{z} \in \Sigma_{\mathbf{x}_1,\mathbf{x}_2} | \nexists \mathbf{u} \in \Sigma_{\mathbf{x}_1,\mathbf{x}_2} : f(\mathbf{u}) \prec f(\mathbf{z})\}$

The flooding algorithm can be modified in a way that incomparable elements are not considered as neighbors ('moves to incomparable solutions are disallowed'). The flooding algorithm may then lead to a forest instead of a tree. For examples (multicriteria knapsack problem, RNA folding) and further discussion of how to efficiently implement this algorithm the reader is referred to [76].

A barrier tree for a continuous landscape is drawn in Fig. 3.10. In this case the saddle points correspond to local maxima. For continuous landscapes the concept of barrier trees can be generalized, but the implementation of flooding algorithms is more challenging due to the infinite number of points that need to be considered. Discretization could be used to get a rough impression of the landscape's geometry.

3.8 Shapes of Pareto Fronts

An interesting, since very general, questions could be: How can the geometrical shapes of Pareto fronts be classified. We will first look at some general descriptions used in literature on how to define the Pareto front w.r.t. convexity and connectedness. To state definitions in an unambiguous way we will make use of Minkowski sums and cones as defined in chapter 2.

[1] here we think of minimization of the objectives.

Algorithm 2 Flooding algorithm

Let $x^{(1)}, \ldots, x^{(N)}$ denote the elements of the search space sorted in ascending order.
$i \to 1; \mathcal{B} = \emptyset$
while $i \leq N$ **do**
  if $N(x_i) \cap \{x^{(1)}, \ldots, x^{(i-1)}\} = \emptyset$ [i. e., $x^{(i)}$ has no neighbour that has been processed.] **then**
    $\{x^{(i)}$ is local minimum$\}$
    Draw $x^{(i)}$ as a new leaf representing basin $B(x^{(i)})$ located at the height of f in the 2-D diagram
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{B(x^{(i)})\}$ $\{$Update set of basins$\}$
  else
    Let $\mathcal{T}(x^{(i)}) = \{B(x^{(i_1)}), \ldots, B(x^{(i_N)})\}$ be the set of basins $B \in \mathcal{B}$ with $N(x^{(i)}) \cap B \neq \emptyset$.
    if $|\mathcal{T}(x^{(i)})| = 1$ **then**
      $B(x^{(i_1)}) \leftarrow B(x^{(i_1)}) \cup \{x^{(i)}\}$
    else
      $\{x^{(i)}$ is a saddle point$\}$
      Draw $x^{(i)}$ as a new branching point connecting the nodes for $B(x^{(i_1)}), \ldots, B(x^{(i_N)})$. Annotate saddle point node with $B(x^{(i)})$ and locate it at the height of f in the 2-D diagram
      $\{$Update set of basins$\}$
      $B(x^{(i)}) = B(x^{(i_1)}) \cup \cdots \cup B(x^{(i_N)}) \cup \{x^{(i)}\}$
      Remove $B(x^{(i_1)}), \ldots, B(x^{(i_N)})$ from $\mathcal{B}$
      $\mathcal{B} \leftarrow \mathcal{B} \cup \{B(x^{(i)})\}$
    end if
  end if
end while

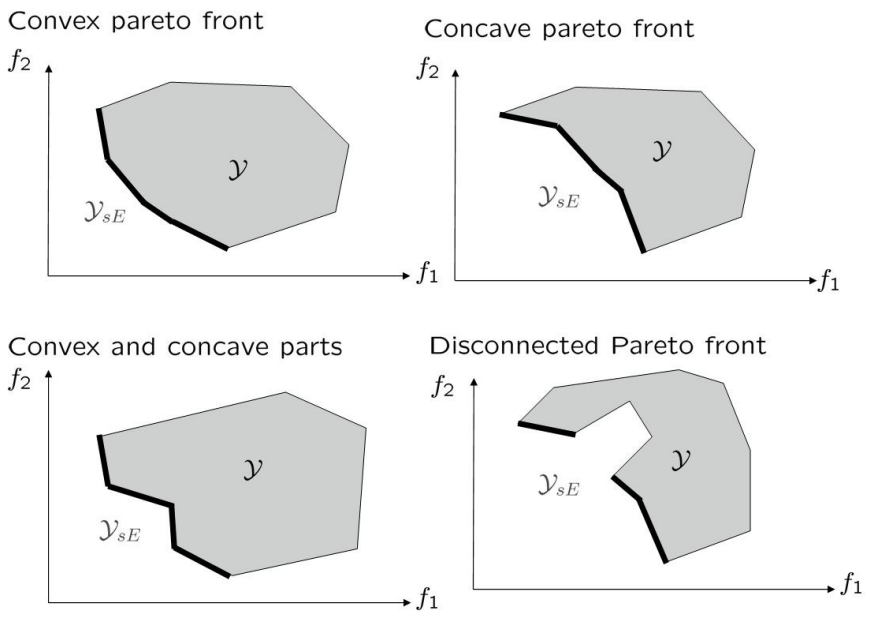

Figure 3.11: Different shapes of Pareto fronts for bi-criteria problems.

Definition 58 *A set $Y \subseteq \mathbb{R}^m$ is said to be* cone convex *w.r.t. the positive orthant, iff $\mathcal{Y}_N + \mathbb{R}^m_{\geq}$ is a convex set.*

Definition 59 *A set $Y \subseteq \mathbb{R}^m$ is said to be* cone concave *w.r.t. the positive orthant, iff $\mathcal{Y}_N - \mathbb{R}^m_{\geq}$ is a convex set.*

Definition 60 *A Pareto front $\mathcal{Y}_N$ is said to be convex (concave), iff it is cone convex (concave) w.r.t. the positive orthant.*

Note, that Pareto fronts can be convex, concave, or may consist of cone convex and cone concave parts w.r.t. the positive orthant.

Convex Pareto fronts allow for better compromise solutions than concave Pareto fronts. In the ideal case of a convex Pareto front, the Pareto front consists only on a single point which is optimal for all objectives. In this situation the decision maker can choose a solution that satisfies all objectives at the same time. The most difficult situation for the decision maker arises, when the Pareto front consists of a separate set of points, one point for each single objective, and these points are separate and very distant from each other. In such a case the decision maker needs to make a either-or decision.

Another classifying criterion of Pareto fronts is connectedness.

Definition 61 *A Pareto front $\mathcal{Y}_N$ is said to be connected, if and only if for all $\mathbf{y}_1, \mathbf{y}_2 \in \mathcal{Y}_N$ there exists a continuous mapping $\phi : [0,1] \to \mathcal{Y}_N$ with $\phi(0) = \mathbf{y}_1$ and $\phi(1) = \mathbf{y}_2$.*

For the frequently occurring case of two objectives, examples of convex, concave, connected and disconnected Pareto fronts are given in Fig. 3.11.

Two further corollaries highlight general characteristics of Pareto-fronts:

Lemma 62 *Dimension of the Pareto front*

Pareto fronts for problems with m-objectives are subsets or equal to $m-1$-dimensional manifolds.

Lemma 63 *Functional dependencies in the Pareto front*

Let $\mathcal{Y}_N$ denote a Pareto front for some multiobjective problem. Then for any sub-vector in the projection to the coordinates in $\{1, \ldots, m\}$ without i, the value of the i-th coordinate is uniquely determined.

Example For a problem with three objectives the Pareto front is a subset of a 2-D manifold that can be represented as a function from the values of the

- first two objectives to the third objective.
- the first and third objective to the second objective
- the last two objectives to the first objective

3.9 Conclusions

In this chapter the focus was on the different types of landscapes, shapes of optima in multiobjective optimization. A general technique to visualize and analyse multicriteria landscapes based on level sets was provided, which also provides insights into the solution of linear programming problems. Moreover, with emphasize on discrete landscapes, the structure of multicriteria landscapes was discussed. Here the barrier tree can be used to figure out, how landscapes are decomposed into valleys and hills and how these are separated from each other. Finally, a classification of different shapes of Pareto fronts was discussed and some general characteristics of such sets were highlighted.

So far, the definitions we used did not exploit differentiability. If differentiability is provided the search for Pareto optima can be supported by powerful theorems as will be discussed in the following chapter.

Chapter 4

Optimality conditions for differentiable problems

In the finite discrete case local optimality of a point $x \in \mathcal{X}$ can be done by comparing it to all neighboring solutions. In the continuous case this is not possible, For differentiable problems we can state conditions for local optimality. We will start with looking at unconstrained optimization, then provide conditions for optimization with equality and inequality constraints and, thereafter, their extensions for the multiobjective case.

4.1 Linear approximations

A general observation we should keep in mind when understanding optimization conditions for differentiable problems is that continuously differentiable functions $f : \mathbb{R}^n \to \mathbb{R}$ can be locally approximated at any point $\mathbf{x}^{(0)}$ by means of linear approximations $f(\mathbf{x}^{(0)}) + \nabla f(\mathbf{x}^{(0)})(\mathbf{x} - \mathbf{x}_0)$ with $\nabla f = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})^\top$. In other words:

$$\lim_{\mathbf{x} \to \mathbf{x}_0} f(\mathbf{x}) - [f(\mathbf{x}_0) + \nabla f(\mathbf{x})(\mathbf{x} - \mathbf{x}_0)] = 0 \tag{4.1}$$

The gradient $\nabla f(\mathbf{x}^{(0)})$ points in the direction of steepest ascent and is orthogonal to the level curves $\mathcal{L}_=(f(\hat{\mathbf{x}}))$ at the point $\hat{\mathbf{x}}$. This has been visualized in Fig. 4.1.

4.2 Unconstrained Optimization

For the unconstrained minimization

$$f(\mathbf{x}) \to \min \tag{4.2}$$

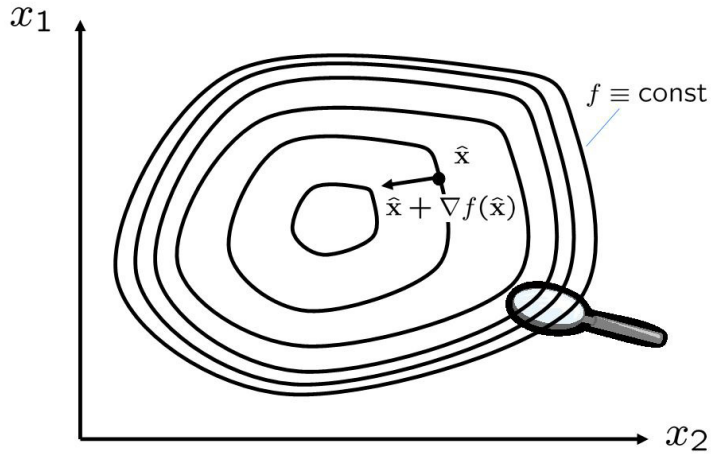

Figure 4.1: Level curves of a continuously differentiable function. Locally the function 'appears' to be a linear function with parallel level curves. The gradient vector $\nabla f(\hat{\mathbf{x}})$ is perpendicular to the local direction of the level curves at $\hat{\mathbf{x}}$.

problem, a well known result from calculus is:

Theorem 64 *Fermat's condition*
Given a differentiable function f. Then $\nabla f(\mathbf{x}^*) = 0$ is a necessary condition for $\mathbf{x}^*$ to be a local extremum. Points with $\nabla f(\mathbf{x}^*) = 0$ are called stationary points. A sufficient condition for $\mathbf{x}^*$ to be a (strict) local minimum is given, if in addition the Hessian matrix $\nabla^2 f(\mathbf{x}^*)$ is positive (semi)definite.

The following theorem can be used to test wether a matrix is positive (semi)definite:

Theorem 65 *A matrix is positive (semi-)definite, iff all eigenvalues are positive (non-negative).*

Alternatively, we may use local bounds to decide whether we have obtained a local or global optimum. For instance, for the problem $\min_{(x,y)\in\mathbb{R}^2}(x-3)^2 + y^2 + \exp y$ the bound of the function is zero and every argument for which the function reaches the value of zero must be a global optimum. As the function is differentiable the global optimum will be also be one of the stationary points. Therefore we can find the global optimum in this case by looking at all stationary points. A more general way of looking at boundaries in the context of optimum seeking is given by the Theorem of Weierstrass discussed in [11]. This theorem is also useful for proving the existence of an optimum. This is discussed in detail in [15].

Theorem 66 *Theorem of Weierstrass*
Let $\mathcal{X}$ be some closed[1] *and bounded subset of $\mathbb{R}^n$, let $f : \mathcal{X} \to \mathbb{R}$ denote a continuous function. Then f attains a global maximum and minimum in $\mathcal{X}$, i. e. $\exists \mathbf{x}_{\min} \in \mathcal{X} : \forall \mathbf{x}' \in \mathcal{X} : f(\mathbf{x}_{\min}) \leq f(\mathbf{x}')$ and $\exists \mathbf{x}_{\max} \in \mathcal{X} : \forall \mathbf{x}' \in \mathcal{X} : f(\mathbf{x}_{\max}) \geq f(\mathbf{x}')$.*

4.3 Equality Constraints

By introducing Lagrange multipliers, theorem 64 can be extended to problems with *equality* constraints, i. e.:

$$f(\mathbf{x}) \to \min, \text{ s.t. } g_1(\mathbf{x}) = 0, \ldots, g_m(\mathbf{x}) = 0 \tag{4.3}$$

In this case the following theorem holds:

Theorem 67 *Let f and $g_1, \ldots, g_m$ denote differentiable functions. Then a necessary condition for $\mathbf{x}^*$ to be a local extremum is given, if there exist multipliers $\lambda_1, \ldots, \lambda_{m+1}$ with at least one $\lambda_i \neq 0$ for $i = 1, \ldots, m$ such that $\lambda_1 \nabla f(\mathbf{x}^*) + \sum_{i=2}^{m+1} \lambda_i \nabla g(\mathbf{x}^*) = \mathbf{0}$.*

For a rigorous proof of this theorem we refer to [15]. Let us remark, that the discovery of this theorem by Lagrange preceded its proof by one hundred years [15].

Next, by means of an example we will provide some geometric intuition for this theorem. In Fig. 4.2 a problem with a search space of dimension two is given. A single objective function f has to be maximized, and the sole constraint function $g_1(\mathbf{x})$ is set to 0.

Let us first look at the level curve $f \equiv -13$. This curve does not intersect with the level curve $g \equiv 0$ and thus there is no feasible solution on this curve. Next, we look at $f \equiv -15$. In this case the two curve intersects in two points with $g \equiv 0$. However, these solutions are not optimal. We can do better by moving to the point, where the level curve of $f \equiv c$ 'just' intersects with $g \equiv 0$. This is the tangent point $\mathbf{x}^*$ with $c = f(\mathbf{x}^*) = -14$.

The tangential point satisfies the condition that the gradient vectors are collinear to each other, i.e. $\exists \lambda \neq 0 : \lambda \nabla g(\mathbf{x}^*) = \nabla f(\mathbf{x}^*)$. In other words, the tangent line to the f level curve at a touching point is equal to the tangent line to the $g \equiv 0$ level curve. Equality of tangent lines is equivalent to the fact that the gradient vectors are collinear.

Another way to reason about the location of optima is to check for each point on the constraint curve whether it can be locally improved or not. For points where the level curve of the objective function intersects with the constraint function, we

[1] Roughly speaking, a closed set is a set which includes all points at its boundary.

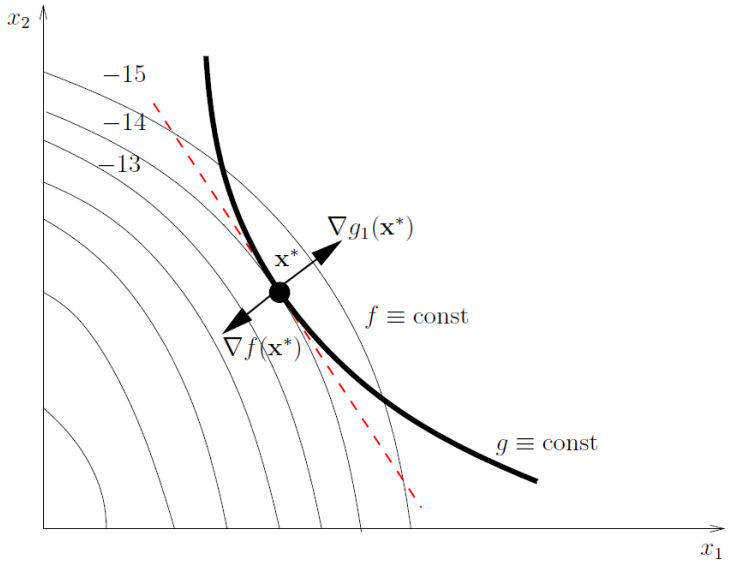

Figure 4.2: Lagrange multipliers: Level-sets for a single objective and single active constraint and search space $\mathbb{R}^2$.

consider the local linear approximation of the objective function. In case of non-zero gradients, we can always improve the point further. In case of zero gradients we already fulfill conditions of the theorem by setting $\lambda_1 = 1$ and $\lambda_i = 0$ for $i = 2, \ldots, m+1$. This way we can exclude all points but the tangential points and local minima of the objective function (unconstrained) from consideration.

In practical optimization often λ_1 is set to 1. Then the equations in the lagrange multiplier theorem boil down to an equation system with $m + n$ unknowns and $m + n$ equations and this gives rise to a set of candidate solutions for the problem. This way of solving an optimization problem is called the *Lagrange multiplier rule*.

Example Consider the following problem:

$$f(x_1, x_2) = x_1^2 + x_2^2 \to \min \tag{4.4}$$

, with equality constraint

$$g(x_1, x_2) = x_1 + x_2 - 1 = 0 \tag{4.5}$$

Due to the theorem of 67, iff $(x_1, x_2)^\top$ is a local optimum, then there exist λ_1 and λ_2 with $(\lambda_1, \lambda_2) \neq (0, 0)$ such that the constraint in equation 4.5 is fulfilled and

$$\lambda_1 \frac{\partial f}{\partial x_1} + \lambda_2 \frac{\partial g}{\partial x_1} = 2\lambda_1 x_1 + \lambda_2 = 0 \tag{4.6}$$

$$\lambda_1 \frac{\partial f}{\partial x_2} + \lambda_2 \frac{\partial g}{\partial x_2} = 2\lambda_1 x_2 + \lambda_2 = 0 \tag{4.7}$$

Let us first examine the case $\lambda_1 = 0$. This entails:

$$\lambda_2 = 0 \tag{4.8}$$

This contradicts the condition that $(\lambda_1, \lambda_2) \neq (0, 0)$.

We did not yet prove, that the solution we found is also a *global* optimum. In order to do so we can invoke Weierstrass theorem, by first reducing the problem to a problem with a reduced search space, say:

$$f_{|A} \to \min \tag{4.9}$$

$$A = \{(x_1, x_2) || x_1| \leq 10 \text{ and } |x_2| \leq 10 \text{ and } x_1 + x_2 - 1 = 0\} \tag{4.10}$$

For this problem a global minimum exists, due to the Weierstrass theorem (the set A is bounded and closed and f is continuous). Therefore, the original problem also has a global minimum in A, as for points outside A the function value is bigger than 199 and in A there are points $x \in A$ where $f(x_1, x_2) < 199$. The (necessary) Lagrange conditions, however, are only satisfied for one point in $\mathbb{R}^2$ which consequently must be the only local minimum and thus it is the global minimum.

Now we consider the case $\lambda_1 = 1$. This leads to the conditions:

$$2x_1 + \lambda_2 = 0 \tag{4.11}$$

$$2x_2 + \lambda_2 = 0 \tag{4.12}$$

and hence $x_1 = x_2$. From the equality condition we get: From the constraint it follows $x_1 + x_1 = 1$, which entails $x_1 = x_2 = \frac{1}{2}$.

Another possibility to solve this problem is by means of substitution: $x_1 = 1 - x_2$ and the objective function can then be written as $f(1-x_2, x_2) = (1-x_2)^2 + x_2^2$. Now minimize the unconstrained 'substitute' function $h(x_2) = (1 - x_2)^2 + x_2^2$. $\frac{\partial h}{x_2} = -2(1 - x_2) + 2x_2 = 0$. This yields $x_2 = \frac{1}{2}$. The second derivative $\frac{\partial^2 f}{\partial^2 x_2} = 4$. This means that the point is a local minimum.

However, not always all candidate solutions for local optima are captured this way as the case $\lambda_1 = 0$ may well be relevant. Brinkhuis and Tikhomirov [15] give an example of such a 'bad' case:

Example Apply the multiplier rule to $f_0(x) \to \min, x_1^2 - x_2^3 = 0$: The Lagrange equations hold at $\hat{x}$ with $\lambda_0 = 0$ and $\lambda_1 = 1$. An interesting observation is that the level curves are cusp in this case at $\hat{x}$, as visualized in Fig. 4.3.

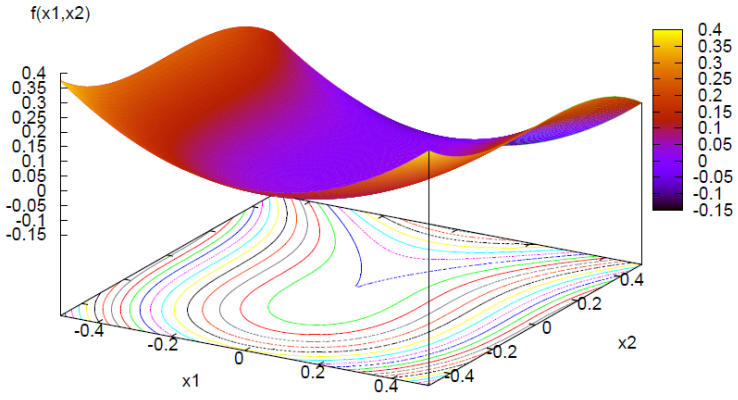

Figure 4.3: The level curves of $x_1^2 - x_2^3$. The level curve through $(0,0)^T$ is cusp.

4.4 Inequality Constraints

For *inequality* constraints the Karush Kuhn Tucker conditions are used as optimality criterion:

Theorem 68 *The Karush Kuhn Tucker conditions are said to hold for* $\mathbf{x}^*$*, if there exist multipliers* $\lambda_1 \geq 0, \ldots, \lambda_{m+1} \geq 0$ *and at least one* $\lambda_i > 0$ *for* $i = 1, \ldots, m+1$*, such that:*

$$\lambda_1 \nabla f(\mathbf{x}^*) + \sum_{i=1}^{m} \lambda_{i+1} \nabla g_i(\mathbf{x}^*) = \mathbf{0} \tag{4.13}$$

$$\lambda_{i+1} g_i(\mathbf{x}^*) = 0, i = 1, \ldots, m \tag{4.14}$$

Theorem 69 *Karush Kuhn Tucker Theorem (Necessary conditions for smooth, convex programming:)*
Assume the objective and all constraint functions are convex in some ϵ*-neighborhood of* $\mathbf{x}^*$*, If* $\mathbf{x}^*$ *is a local minimum, then there exiss* $\lambda_1, \ldots, \lambda_{m+1}$ *such that KKT conditions are fulfilled.*

Theorem 70 *The KKT conditions are sufficient for optimiality, provided* $\lambda_1 = 1$*. In this case* $\mathbf{x}^*$ *is a local minimum.*

Note that if $\mathbf{x}^*$ is in the interior of the feasible region (a Slater point), all $g_i(\mathbf{x}) < 0$ and thus $\lambda_1 > 0$.

The next examples discuss the usage of the Karush Kuhn conditions:

Example In order to get familiar with the KKT theorem we apply it to a very simple situation (solvable also with high school mathematics). The task is:

$$1 - x^2 \to \min, x \in [-1, 3]^2 \tag{4.15}$$

First, write the task in its standard form:

$$f(x) = 1 - x^2 \to \min \tag{4.16}$$

subject to constraints

$$g_1(x) = -x - 1 \leq 0 \tag{4.17}$$

$$g_2(x) = x - 3 \leq 0 \tag{4.18}$$

The existence of the optimum follows from Weierstrass theorem, as (1) the feasible subspace [-1,3] is bounded and closed and (2) the objective function is continuous.

The KKT conditions in this case boil down to: There exists $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}_0^+$ and $\lambda_3 \in \mathbb{R}_0^+$ and $(\lambda_1, \lambda_2, \lambda_3) \neq (0, 0, 0)$ such that

$$\lambda_1 \frac{\partial f}{\partial x} + \lambda_2 \frac{\partial g_1}{\partial x} + \lambda_3 \frac{\partial g_1}{\partial x} = -2\lambda_1 x - \lambda_2 + \lambda_3 = 0 \tag{4.19}$$

$$\lambda_2(-x - 1) = 0 \tag{4.20}$$

$$\lambda_3(x - 3) = 0 \tag{4.21}$$

.

First, let us check whether $\lambda_1 = 0$ can occur:

In this case the three equations (4.19, 4.20, and 4.21) will be:

$$-\lambda_2 + \lambda_3 = 0 \tag{4.22}$$

$$\lambda_2(-x - 1) = 0 \tag{4.23}$$

$$\lambda_3(x - 3) = 0 \tag{4.24}$$

.

and $(\lambda_2, \lambda_3) \neq (0, 0)$, and $\lambda_i \geq 0, i = 2, 3$. From 4.22 we see that $\lambda_2 = \lambda_3$. By setting $\lambda = \lambda_2$ we can write

$$\lambda(-x - 1) = 0 \tag{4.25}$$

and

$$\lambda(x - 3) = 0 \tag{4.26}$$

for the equations 4.23 and 4.24. Moreover $\lambda \neq 0$, for $(\lambda, \lambda) = (\lambda_2, \lambda_3) \neq (0, 0)$. From this we get that $-x - 1 = 0$ and $x - 3 = 0$. Which is a contradiction so the case $\lambda_1 = 0$ cannot occur – later we shall see that this could have derived by using

a theorem on Slater points 70.
Next we consider the case $\lambda_1 \neq 0$ (or equivalently $\lambda_1 = 1$): In this case the three equations (4.19, 4.20, and 4.21) will be:

$$-2x - \lambda_2 + \lambda_3 = 0 \tag{4.27}$$

,

$$\lambda_2(-x - 1) = 0 \tag{4.28}$$

, and

$$\lambda_3(x - 3) = 0 \tag{4.29}$$

We consider four subcases:

case 1: $\lambda_2 = \lambda_3 = 0$. This gives rise to $x = 0$

case 2: $\lambda_2 = 0$ and $\lambda_3 \neq 0$. In this case we get as a condition on x: $2x(x-3) = 0$ *and* $x \neq 0$ or equivalently $x = 3$

case 3: $\lambda_2 \neq 0$ and $\lambda_3 = 0$. We get from this: $-2x(-x-1) = 0$ *and* $x \neq 0$ or equivalently $x = -1$.

case 4: $\lambda_2 \neq 0$ and $\lambda_3 \neq 0$. This cannot occur as this gives rise to $-x - 1 = 0$ *and* $x - 3 = 0$ (contradictory conditions).

In summary we see that a maximum can possibly only occur in $x = -1$, $x = 0$ or $x = 3$. By evaluating f on these three candidates, we see that f attains its global minimum in $x = 3$ and the value of the global minimum is -8. Note that we invoked also the Weierstrass theorem in the last conclusion: the Weierstrass theorem tells us that the function f has a global minimum in the feasible region ([-1.3]) and KKT (necessary conditions) tell us that it must be one of the three above mentioned candidates.

4.5 Multiple Objectives

For a recent generalization of the Lagrange multiplier rule to multiobjective optimization we refer to [38]. For multicriterion optimisation the KKT conditions can be generalized as follows:

Theorem 71 *Fritz John necessary conditions*
A neccessary condition for $\mathbf{x}^*$ *to be a locally efficient point is that there exist vectors* $\lambda_1, \ldots, \lambda_k$ *and* $\upsilon_1, \ldots, \upsilon_m$ *such that*

$$\lambda \succ \mathbf{0}, \upsilon \succ \mathbf{0} \tag{4.30}$$

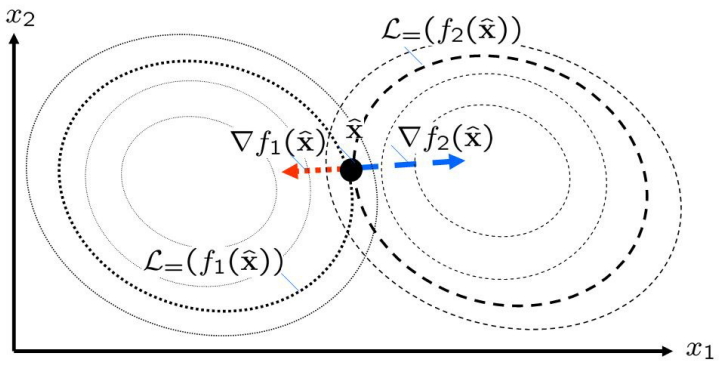

Figure 4.4: Level curves of the two objectives touching in one point indicate locally Pareto optimal points in the bi-criterion case, provided the functions are differentiable.

$$\sum_{i=1}^{k} \lambda_i \nabla f_i(\mathbf{x}^*) - \sum_{i=1}^{m} \upsilon_i \nabla g_i(\mathbf{x}^*) = \mathbf{0}. \tag{4.31}$$

$$\upsilon_i g_i(\mathbf{x}^*) = 0, i = 1, \ldots, m \tag{4.32}$$

A sufficient condition for points to be Pareto optima follows:

Theorem 72 *Karush Kuhn Tucker sufficient conditions for a solution to be Pareto optimal:*
Let $\mathbf{x}^*$ *be a feasible point. Assume that all objective functions are locally convex and all constraint functions are locally concave, and the Fritz John conditions hold in* $\mathbf{x}^*$*, then* $\mathbf{x}^*$ *is a local efficient point.*

In the unconstrained case we get the simple condition:

Corollary 73 *In the unconstrained case Fritz John neccessary conditions reduce to*

$$\lambda \succ \mathbf{0} \tag{4.33}$$

$$\sum_{i=1}^{k} \lambda_i \nabla f_i(\mathbf{x}^*) = \mathbf{0}. \tag{4.34}$$

In 2-dimensional spaces this criterion reduces to the observation, that either one of the objectives has a zero gradient (neccesary condition for ideal points) or the gradients are collinear as depicted in Fig. 4.4. A detailed description of the conditions for multiobjective optimization is given in [62].

Chapter 5

Scalarization Methods

A straightforward idea to recast a multiobjective problem as a single objective problem is to sum up the objectives in an weighted sum and then to maximize/minimize the weighted sum of objectives. More general is the approach to aggregate the objectives to a single objective by a so-called utility function, which does not have to be a linear sum but usually meets certain monotonicity criteria. Techniques that sum up multiple objectives into a single one by mean of an aggregate function are termed *scalarization techniques*. A couple of questions arise when applying such techniques:

- Does the global optimization of the aggregate function always (or in certain cases) result in an efficient point?
- Can all solutions on the Pareto front be obtained by varying the (weight) parameters of the aggregate function?
- Given that the optimization of the aggregate leads to an efficient point, how does the choice of the weights control the position of the obtained solution on the Pareto front?

Section 5.1 starts with linear aggregation (weighted sum) and answers the aforementioned questions for it. The insights we gain from the linear case prepare us for the generalization to nonlinear aggregation in Section 5.2. The expression or modeling of preferences by means of of aggregate functions is a broad field of study called Multi-attribute utility theory (MAUT). An overview and examples are given in Section 5.3. A common approach to solve multicriteria optimization problems is the distance to a reference point method. Here the decision pointer defines an desired 'utopia' point and minimizes the distance to it. In Section 5.4 we will discuss this method as a special case of a scalarization technique.

5.1 Linear Aggregation

Linear weighting is an straightforward way to summarize objectives. Formally the problem:

$$f_1(x) \to \min, \ldots, f_m(x) \to \min \tag{5.1}$$

is replaced by:

$$\sum_{i=1}^{m} w_i f_i(x) \to \min, w_1, \ldots, w_m > 0 \tag{5.2}$$

A first question that may arise is, whether the solution of problem 5.2 is an efficient solution of problem 5.1. This is indeed the case as points that are non-dominated w.r.t. problem 5.1 are also non-dominated w.r.t. problem 5.2, which follows from:

$$\forall \mathbf{y}^{(1)}, \mathbf{y}^{(2)} \in \mathbb{R}^m : \mathbf{y}^{(1)} \prec \mathbf{y}^{(2)} \Rightarrow \sum_{i=1}^{m} y_i^{(1)} < \sum_{i=1}^{m} y_i^{(2)} \tag{5.3}$$

Another question that arises is, whether we can find all points on the Pareto front using linear aggregation and varying the weights or not. The following theorem provides the answer. To state the the theorem, we need the following definition:

Definition 74 *Proper efficiency [28]*
Given a Pareto optimization problem (Eq. 5.1), then a solution x is called efficient in the Geoffrion sense or properly efficient*, iff (a) it is efficient, and (b) there exists a number $M > 0$ such that $\forall i = 1, \ldots, m$ and $\forall x \in \mathcal{X}$ satisfying $f_i(x) < f_i(x^*)$, there exists an index j such that $f_j(x^*) < f_j(x)$ and:*

$$\frac{f_i(x^*) - f_i(x)}{f_j(x) - f_j(x^*)} \leq M$$

The image of a properly efficient point we will term properly non-dominated. The set of all proper efficient points is termed proper efficient set*, and its image* proper Pareto front*.*

Note, that in the bi-criterion case, the efficient points which are Pareto optimal in the Geoffrion sense are those points on the Pareto-front, where the slope of the Pareto front (f_2 expressed as a function of f_1) is finite and non-zero (see Fig. 5.1). The parameter M is interpreted as trade-off. The proper Pareto optimal points can thus be viewed as points with a bounded tradeoff.

Theorem 75 *Weighted sum scalarization*
Let us assume a Pareto optimization problem (Eq. 5.1) with a Pareto front that

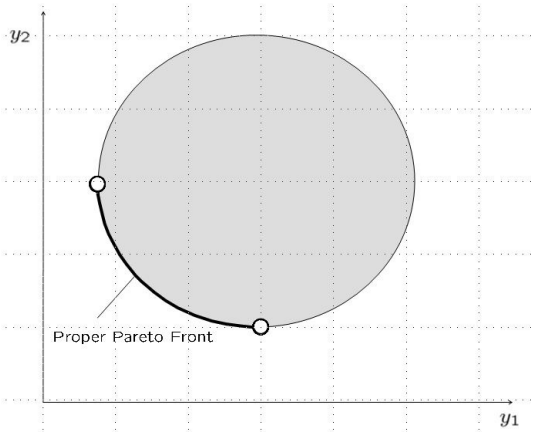

Figure 5.1: The proper Pareto front for a bicriteria problem, for which in addition to many proper Pareto optimal solutions there exist also two non-proper Pareto optimal solutions.

is cone convex w.r.t. positive orthant ($\mathbb{R}^m_{\geq}$). Then for each properly efficient point $x \in \mathcal{X}$ there exist weights $w_1 > 0, \ldots, w_m > 0$ such that x is one of the solutions of $\sum_{i=1}^m f_i(x) \to \min$.

In case of problems with a non-convex pareto front it is not always possible to find weights for a given proper efficient point x such that x is one of the solutions of $\sum_{i=1}^m f_i(x) \to \min$. A counterexample is given in the following example:

Example In Fig. 5.2 the Pareto fronts of two different bi-criterion problems are shown. The figure on the right hand side shows a Pareto front which is cone convex with respect to the positive orthant. Here the tangential points of the level curves of $w_1 y_1 + w_2 y_2$ are the solutions obtained with linear aggregation. Obviously, by changing the slope of the level curves by varying one (or both) of the weights, all points on the Pareto front can be obtained (and no other). On the other hand, for the concave Pareto front shown on the right hand side only the extreme solutions at the boundary can be obtained.

As the example shows linear aggregation has a tendency to obtain extreme solutions on the Pareto front, and its use is thus problematic in cases where no a-priori knowledge of the shape of the Pareto front is given. However, there exist aggregation functions which have less tendency to obtain extreme solutions or even allow to obtain all Pareto optimal solutions. They will be discussed in the next section.

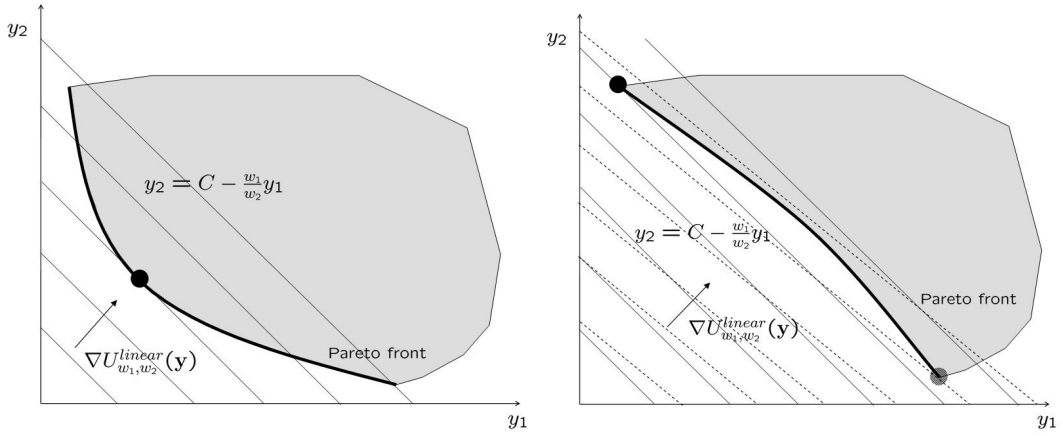

Figure 5.2: The concave (left) and convex Pareto front (right).

5.2 Nonlinear Aggregation

Instead of linear aggregation we can use nonlinear aggregation approaches, e.g. compute a product of the objective function value. The theory of utility functions can be viewed as a modeling approach for (non)linear aggregation functions.

A utility function assigns to each combination of values that may occur in the objective space a scalar value - the so-called utility. This value is to be maximized. Note that the linear aggregation was to be minimized. Level curves of the utility function are interpreted as indifference curves (see Fig. 5.3).

In order to discuss a scalarization method it may be interesting to analyze where on the Pareto front the Pareto optimal solution that is found by maximizing the utility function is located. Similar to the linear weighting function discussed earlier, this is the point where the level curves of the utility (looked upon in descending order) first intersect with the Pareto front (see Fig. 5.4).

5.3 Multi-Attribute Utility Theory

Next, we will discuss a concrete example for the design of a utility function. This example will illustrate many aspects of how to construct utility functions in a practically useful, consistent, and user-friendly way.

Example Consider you want to buy a car. Then you may focus on three objectives: speed, price, fuel-consumption. These three criteria can be weighted. It is often not wise to measure the contribution of an objective function to the overall utility in a linear way. A elegant way to model it is by specifying a function that measures the degree of satisfaction. For each possible value of the objective

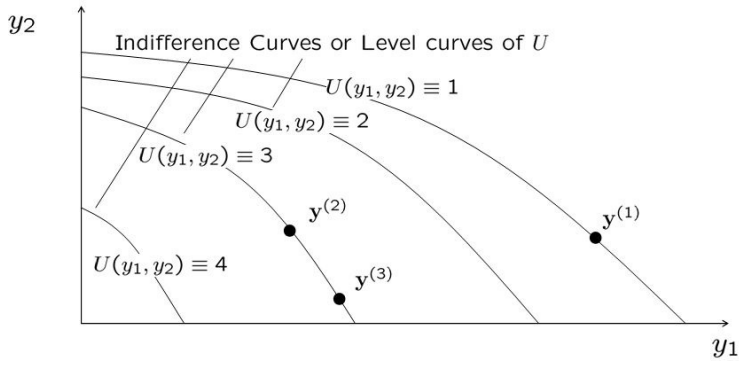

Figure 5.3: Utility function for a bi-criterion problem. If the decision-maker has modeled this utility function in a proper way, he/she will be indifferent whether to choose $\mathbf{y}^{(2)}$ and $\mathbf{y}^{(3)}$, but prefer $\mathbf{y}^{(3)}$ and $\mathbf{y}^{(2)}$ to $\mathbf{y}^{(1)}$.

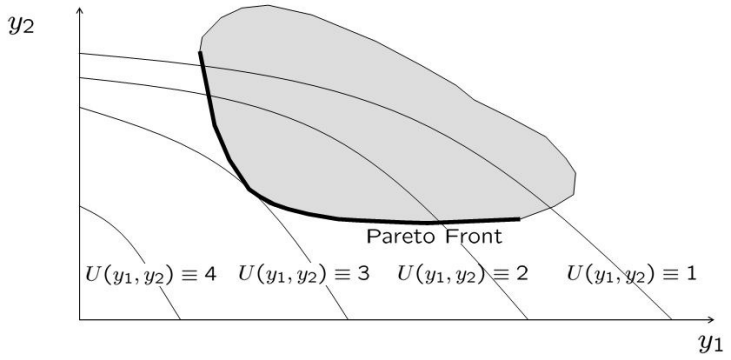

Figure 5.4: The tangential point of the Pareto front with the indifference curves of the utility function U here determines where the solution of the maximization of the utility function lies on the Pareto front.

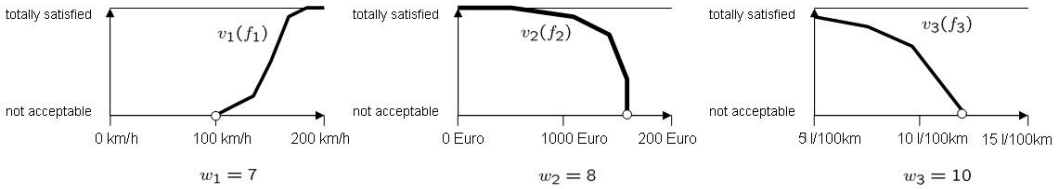

Figure 5.5: The components (value functions) of a multiattribute utility function.

function we specify the degree of satisfaction of this solution on a scale from 0 to 10 by means of a so-called *value function*. In case of speed, we may demand that a car is faster than 80m/mph but beyond a speed of, say, 180 km/h the increase of our satisfaction with the car is marginal, as we will not have many occasions where driving as this speed gives us advantages. It can also be the case, that the objective is to be minimized. As an example, we consider the price of the car. The budget that we are allowed to spend marks an upper bound for the point where the value function obtains a value of zero. Typically, our satisfaction will grow if the price is decreased until a critical point, where we may no longer trust that the solution is sold for a fair price and we may get suspicious of the offer.

The art of the game is then to sum up these objectives to a single utility function. One approach is as follows: Given value functions $v_i : \mathbb{R} \to [0, 10], i = 1, \ldots, m$ mapping objective function values to degree of satisfaction values, and their weights $w_i, i = 1, \ldots, m$, we can construct the following optimization problem with constraints:

$$U(\mathbf{f}(x)) = \alpha \underbrace{\frac{1}{m} \sum_{i=1}^{m} w_i v_i(f_i(x))}_{\text{common interest}} + \beta \underbrace{\min_{i \in \{1,\ldots,m\}} w_i v_i(f_i(x))}_{\text{minority interest}}, \tag{5.4}$$

$$(\text{here: } m = 3) \tag{5.5}$$

$$s.t.\ v_i(f_i(x)) > 0, i = 1, \ldots, m \tag{5.6}$$

Here, we have one term that looks for the 'common interest'. This term can be comparably high if some of the value functions have a very high value and others a very small value. In order to enforce a more balanced solutions w.r.t. the different value functions, we can also consider to focus on the value function which is least satisfied. In order to discard values from the search space, solution candidates with a value function of zero are considered as infeasible by introducing strict inequality constraints.

A very similar approach is the use of desirability indices. They have been first proposed by Harrington [41] for applications in industrial quality management.

Another well known reference for this approach is [20].

We first give a rough sketch of the method, and then discuss its formal details.

As in the previously described approach, we map the values of the objective function to satisfaction levels, ranging from not acceptable (0) to totally satisfied (1). The values in between 0 and one indicate the grey areas. Piecewise defined exponential functions are used to describe the mappings. They can be specified by means of three parameters. The mapped objective function values are now called desirability indices. Harrington proprosed to aggregate these desirability indices by a product expression, the minimization of which leads to the solution of the multiobjective problem.

The functions used for the mapping of objective function values to desirability indices are categorized into one-sided and two sided functions. Both have a parameter y_i^{min} (lower specification limit), y_i^{max} (upper specification limit), l_i, r_i (shape parameters), and t_i (symmetry center). The one-sided functions read:

$$D_i = \begin{cases} 0, & y_i < y_i^{min} \\ \left(\frac{y_i - y_i^{min}}{t_i - y_i^{min}}\right)^{l_i}, & y_i^{min} < y_i < t_i \\ 1, & y_i \geq t_i \end{cases} \tag{5.7}$$

and the two sided functions read:

$$D_i = \begin{cases} 0, & y_i < y_i^{min} \\ \left(\frac{y_i - y_i^{min}}{t_i - y_i^{min}}\right)^{l_i}, & y_i^{min} \leq y_i \leq t_i \\ \left(\frac{y_i - y_i^{max}}{t_i - y_i^{max}}\right)^{r_i}, & t_i < y_i \leq y_i^{max} \\ 0, & y_i > y^{max} \end{cases} \tag{5.8}$$

The two plots in Fig. 5.6 visualize one-sided (l) and two-sided (r) desirability indexes.

(Error converting to EPS. Consult Lieuwe)

Figure 5.6: In the left figure we see and examples for one-sided desirability function with parameters $y^{min} = -1, y^{max} = 1, l \in \{0.5, 1, 1.5\}$. The left side displays a plot of two sided desirability functions for parameters $y^{min} = -1.y^{max} = 1$, $l \in \{0.5, 1.0, 1.5\}$, and r being set to the same value than l.
The aggregation of the desirability indices is done by means of a product formula, that is to be maximized:

$$D = (\prod_{i=1}^{k} D_i(y_i))^{\frac{1}{k}} \tag{5.9}$$

In literature many approaches for constructing non-linear utility functions are discussed.

The Cobbs Douglas utility function is widely used in economy. Let $f_i, i = 1, \ldots, m$ denote non-negative objective functions, then the Cobbs Douglas utility function reads:

$$U(x) = \prod_{i=1}^{m} f_i(x)^{\alpha_i} \tag{5.10}$$

It is important to note, that for the Cobbs Douglas utility function the objective function values are to be minimized, while the utility is to be maximized. Indeed, the objective function values, the values α_i, and the utility have usually an economic interpretation, such as the amount of goods: f_i, the utility of a combination of goods: U, and the elasticities of demand: α_i. A useful observation is that taking the logarithm of this function transforms it into a linear expression:

$$\log U(x) = \sum_{i=1}^{m} \alpha_i \log f_i(x) \tag{5.11}$$

The linearity can often be exploited to solve problems related to this utility function analytically.

A more general approach for construction of utility functions is the Keeney Raiffa utility function approach [50]. Let f_i denote non-negative objective functions:

$$U(x) = K \prod_{i=1}^{m} (w_i u_i(f_i(x)) + 1) \tag{5.12}$$

Here w_i are weights for the objective functions between 0 and 1 and K denotes a positive scaling constant. Moreover, u_i denote functions that are strictly increasing

for positive input values. A general remark on how to construct utility functions is, that the optimization of these functions should lead to Pareto optimal solutions. This can be verified by checking the monotonicity condition for a given utility function U:

$$\forall x, x' \in \mathcal{X} : x \prec x' \Rightarrow U(x) > U(x') \tag{5.13}$$

This condition can be easily verified for the two given utility functions.

5.4 Distance to a Reference Point Methods

A special class of utility functions is the distance to the reference point (DRP) method. Here the user specifies an ideal solution (or: utopia point) in the objective space. Then the goal is to get as close as possible to this ideal solution. The distance to the ideal solution can be measured by some distance function, for example a weighted Minkowski distance with parameter γ. This is defined as:

$$d(\mathbf{y}, \mathbf{y}') = [\sum_{i=1}^{m} w_i |y_i - y_i'|^{\gamma}]^{\frac{1}{\gamma}}, \gamma \geq 1, w_1 > 0, \ldots, w_m > 0 \tag{5.14}$$

Here, w_i are positive weights that can be used to normalize the objective function values. In order to analyze which solution is found by means of a DRP method we can interpret the distance to the reference point as an utility function (with the utility value to be minimized). The indifference curves in case of $\gamma = 2$ are spheres (or ellipsoids) around the utopia point. For $\gamma > 2$ one obtains different super-ellipsoids as indifference curves. Here, a super-ellipsoid around the utopia point $\mathbf{f}^*$ of radius $r \geq 0$ is defined as a set:

$$S(r) = \{\mathbf{y} \in \mathbb{R}^m | d(\mathbf{y}, \mathbf{f}^*) = r\} \tag{5.15}$$

with $d : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}_0^+$ being a weighted distance function as defined in Eq. 5.14.

Example In Figure 5.7 for two examples of a DRP method it is discussed, how the location of the optimum is obtained geometrically, given the image set $\mathbf{f}(\mathcal{X})$. We look for the super-ellipsoid with the smallest radius that still touches the image set. If two objective functions are considered and weighted Euclidean distance is used, i.e. $\gamma = 2$, then the super-ellipsoids are regular ellipses (Fig. 5.7). If instead a manhattan distance ($\gamma = 1$) is used with equal weights, then we obtain diamond shaped super-ellipsoids (Fig. 5.7).

Not always an efficient point is found when using the DRP method. However, in many practical cases the following sufficient condition can be used in order to make sure that the DRP method yields an efficient point. This condition is summarized in the following lemma:

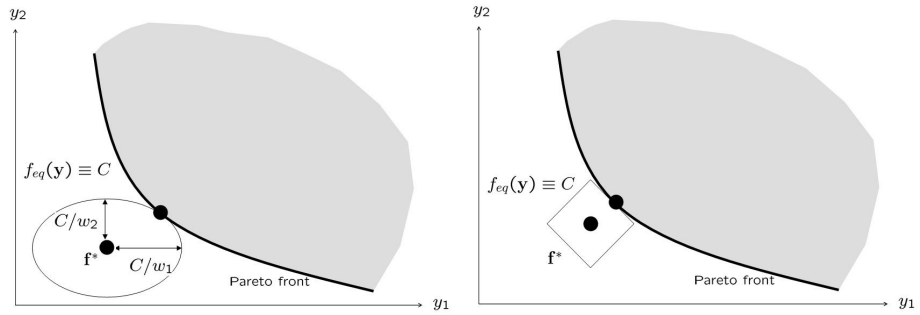

Figure 5.7: Optimal points obtained for two distance to DRP methods, using the weighted Euclidean distance (left) and the manhattan distance (right).

Lemma 76 *Let* $\mathbf{f}^* \in \mathbb{R}^m$ *denote an utopia point, then*

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathcal{X}} d(\mathbf{f}(\mathbf{x}), \mathbf{f}^*) \tag{5.16}$$

is an efficient point, if for all $\mathbf{y} \in \mathbf{f}(\mathcal{X})$ *it holds that* $\mathbf{f}^* \preceq \mathbf{y}$.

Often the utopia point is chosen to be zero (for example when the objective functions are strictly positive). Note that it is neither sufficient nor necessary that $\mathbf{f}^*$ is non-dominated by $\mathbf{f}(\mathcal{X})$. The counterexamples given in Fig. 5.9 confirm this.

Another question that may arise, using the distance to a reference point method is whether it is possible to find all points on the Pareto front, by changing the weighting parameters w_i of the metric. Even in the case that the utopia points dominates all solutions we cannot obtain all points on the Pareto front by minimizing the distance to the reference in case of $\gamma < \infty$. Concave parts of the Pareto front may be overlooked, because we encounter the problems that we discussed earlier in case of linear weighting.

However, in case of the weighted Tschebycheff distance (or: maximum distance)

$$d^\infty_{\mathbf{w}}(\mathbf{y}, \mathbf{y}') = \max_{i\in\{1,\ldots,m\}} w_i|y_i - y'_i| \tag{5.17}$$

we can obtain all points on the Pareto front by optimizing the distance for different weights w_i. In more detail, the following condition is satisfied:

$$\forall \mathbf{y} \in \mathcal{Y}_N : \exists w_1, \ldots, w_m : \mathbf{y} \in \arg\min_{\mathbf{y}'\in\mathcal{Y}} d^\infty_{\mathbf{y}}(\mathbf{y}', \mathbf{f}^*)$$

However, by using the Tschebycheff metric we may also obtain dominated points, even in cases where $\mathbf{f}^*$ dominates all solutions in $\mathbf{f}(\mathcal{X})$. These solutions are then weakly dominated solutions.

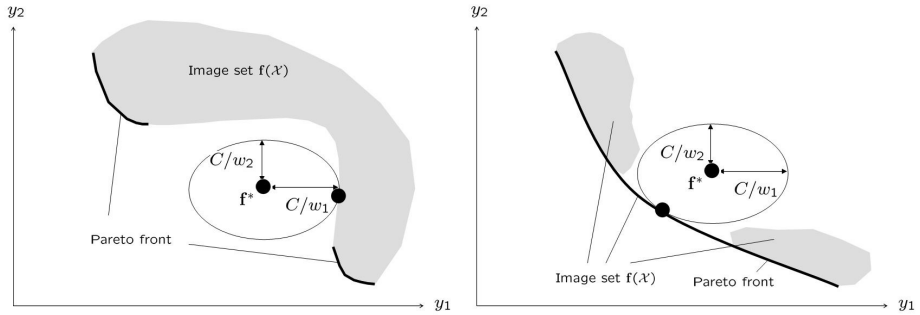

Figure 5.8: In the left figure we see and example for a utopia point which is non-dominated by the image set but the corresponding DRP method does not yield a solution on the Pareto front. In the right figure we see an example where an utopia point is dominated by some points of the image set, but the corresponding DRP method yields a solution on the Pareto front.

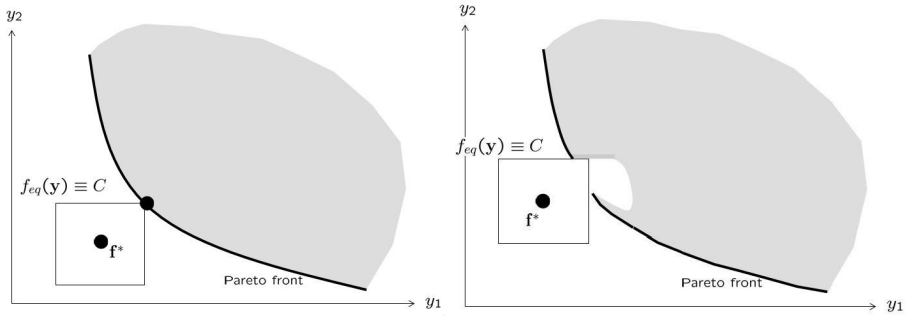

Figure 5.9: In the left figure we see and example where an non-dominated point is obtained using a DRP with the Tschebychev distance. In the right figure we see an example where also dominated solutions minimize the Tschebychev distance to the reference point. In these cases a non-dominated solution may be missed by this DRP method if it returns some single solution minimizing the distance.

In summary, distance to a reference point methods can be seen as an alternative scalarization approach to utility function methods with a clear interpretation of results. They require the definition of a target point (that ideally should dominate all potential solutions), and also a metric needs be specified. We note, that the Euclidean metric is not always the best choice. Typically, the weighted Minkowski metric is used as a metric. The choice of weights for that metric and the choice of γ can significantly influence the result of the method. Except for the Tschebycheff metric, it is not possible to obtain all points on a Pareto front by changing the weights of the different criteria. The latter metric, however, has the disadvantage that also weakly dominated points may be obtained.

Chapter 6

Transforming Multicriteria into Constrained Single-Criterion Problems

This chapter will highlight two common approaches for transforming Multicriteria into Constrained Single-Criterion Problems. In *Compromise Programming* (or ϵ-Constraint Method), $m-1$ of the m objectives are transformed into constraints. Another approach is put forward in the so-called *goal attainment* and *goal programming method*. Here a target vector is specified (similar to the distance to a reference point methods), and a direction is specified. The method searches for the best feasible point in the given direction. For this a costraint programming task is solved.

6.1 Compromise Programming or ϵ-Constraint Methods

In compromise programming we first choose f_1 to be the objective function that has to be solved with highest priority and then re-state the original multicriteria optimization problem (Eq. 1.11):

$$f_1(x) \to \min, f_2(x) \to \min, \ldots, f_m(x) \to \min \tag{6.1}$$

into the single-criterion constrained problem:

$$f_1(x) \to \min, f_2(x) \leq \epsilon_2, \ldots, f_m(x) \to \epsilon_m. \tag{6.2}$$

In figure 6.1 the method is visualized for the bi-criteria case ($m = 2$). Here, it can be seen that if the constraint boundary shares points with the Pareto front,

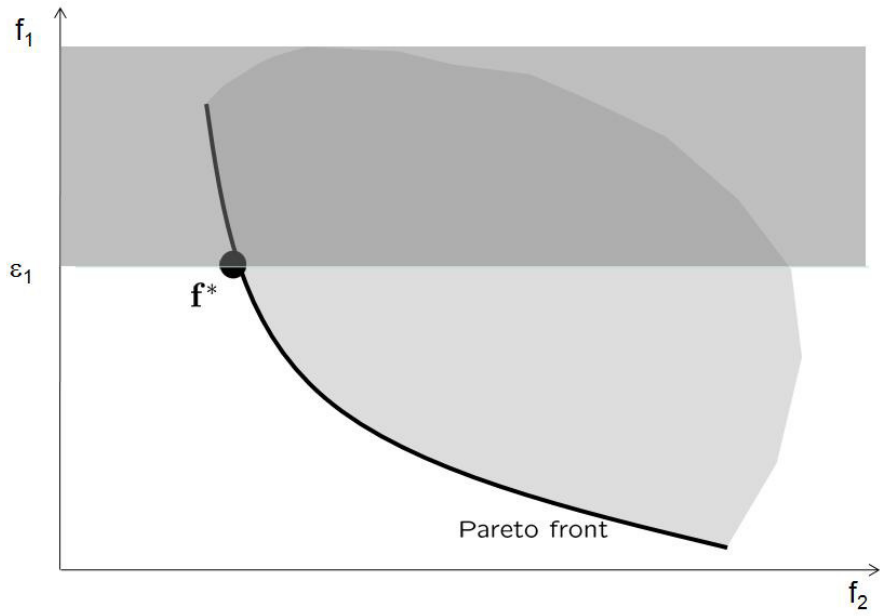

Figure 6.1: Compromise Programming in the bi-criteria case. The second objective is transformed into a constraint.

these points will be the solutions to the problem in Eq. 6.2. Otherwise, it is the solution that is the closest solution to the constraint boundary among all solutions on the Pareto-front. In many cases the solutions are obtained at points x where all objective function values $f_i(x)$ are equal to ϵ_i for $i = 1, \ldots, m$. In these cases, we can obtain optimal solutions using the Lagrange Multiplier method discussed in chapter 4. Not in all cases the solutions obtained with the compromise programming method are Pareto optimal. The method might also find a weakly dominated point, which then has the same aggregated objective function value than some non-dominated point. The construction of an example is left as an exercise to the reader.

The compromise programming method can be used to approximate the Pareto front. For a m dimensional problem a $m-1$ dimensional grid needs to be computed that cover the $m-1$ dimensional projection of the bounding box of the Pareto front. Due to Lemma 63 given $m-1$ coordinates of a Pareto front, the m-th coordinate is uniquely determined as the minimum of that coordinate among all image vectors that have the $m-1$ given coordinates. As an example, in a 3-D case (see Figure 6.2) we can place points on a grid stretching out from the minimal point (f_1^{min}, f_1^{max}) to the maximal point (f_2^{min}, f_2^{max}) . It is obvious that, if the grid resolution is kept constant, the effort of this method grows exponentially with the number of objective functions m.

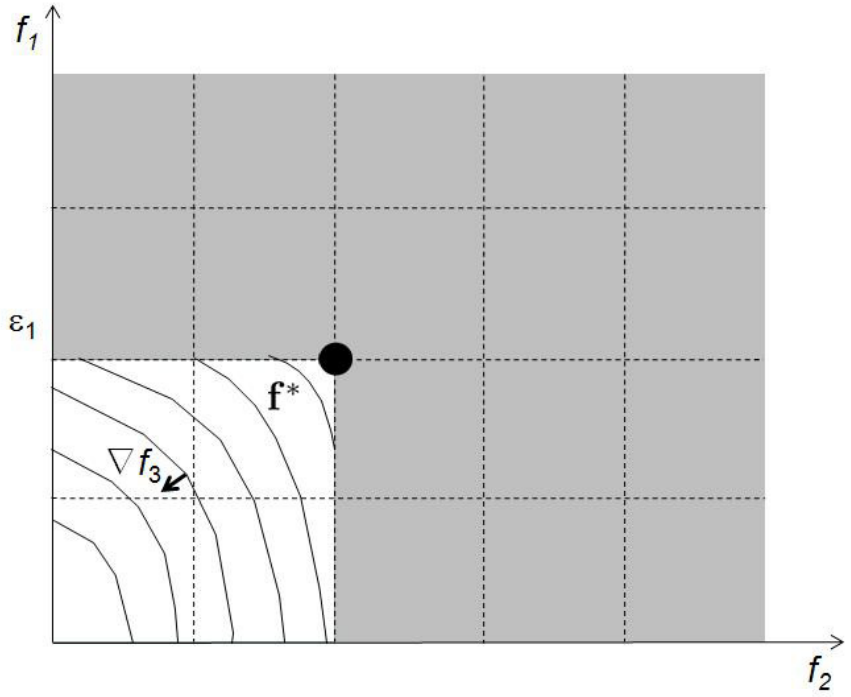

Figure 6.2: Compromised Programming used for approximating the Pareto front with 3 objectives.

This method for obtaining a Pareto front approximation is easier to control than the to use weighted scalarization and change the weights gradually. However, the knowledge of the ideal and the Nadir point is needed to compute the approximation, and the computation of the Nadir point is a difficult problem in itself.

6.2 Concluding remarks on single point methods

In the last two chapters various approaches have been discussed to reformulate a multiobjective problem into a single-objective or a constrained single-objective problem. The methods discussed have in common that they result in a single point, why they also are referred to as *single point methods*.

In addition, all single point methods have parameters the choice of which determines the location of the optimal solution. Each of this methods has, as we have seen, some unique characteristics and it different to give a global comparison of them. However, a criterion that can be assessed for all single point methods is, whether they are always resulting in Pareto optimal solutions. Moreover, we

investigated whether by changing their parameters all points on the Pareto front can be obtained.

To express this in a more formal way we may denote a single point method by a function $A : P \times C \mapsto \mathbb{R}^m \cup \{\Omega\}$, where P denotes the space of multi-objective optimization problems, C denotes the parameters of the method (e.g. the weights in linear weighting). In order to classify a method A we introduce the following two definitions:

Definition 77 *Proper single point method*
A method $\mathtt{A}$ *is called* proper*, if and only if for all* $p \in P$ *and* $c \in C$ *either* p *has a solution and the point* $A(p, c)$ *is Pareto optimal or* p *has no solution and* $\mathtt{A}(p, c) = \Omega$*.*

Definition 78 *Exhaustive single point method*
A method $\mathtt{A}$ *is called* exhaustive *if for all* $p \in P$*:* $\mathcal{Y}_N(p) \subseteq \bigcup_{c \in C} \mathtt{A}(p, c)$*, where* $\mathcal{Y}_N(p)$ *denotes the Pareto front of problem* p*.*

In table **??** a summary of the properties of methods we discussed: In the following chapters on algorithms for Pareto optimization the single point methods often serve as components of methods that compute the entire Pareto front, or an approximation to it.

Single Point Method	Proper	Exhaustive	Remarks
Linear Weighting	Yes	No	Exhaustive for convex Pareto fronts with only proper Pareto optima
Weighted Euclidean DRP	No	No	Proper if reference point dominates all Pareto optimal points
Weighted Tschebyschev DRP	No	Yes	Weakly non-dominated points can be obtained, even when reference point dominates all Pareto optimal points
Desirability index	No	No	The classification of proper/exhaustive is not relevant in this case.
Goal programming	No	No	For convex and concave Pareto fronts with the method is proper and exhaustive if the reference point dominates all Pareto optimal solutions
Compromise programming	No	Yes	In two dimensional objective spaces the method is proper. Weakly dominated points may qualify as solutions for more than three dimensional objective spaces

Table 6.1: Aggregation methods and their properties.

Part II

Algorithms for Pareto Optimization

Chapter 7

Pareto Front Computing with Deterministic Methods

In the previous chapters we looked at ways to reformulate multiobjective optimization problems as single objective (constrained) optimization problems. However, it can be very desirable for the decision maker to know the entire Pareto front.

Methods to compute the Pareto front or an finite set approximation to it can be subdivided into deterministic methods that often guarantee convergence to sets consisting of Pareto points. Some of these methods can also compute approximation sets that distribute in a well defined way, e.g. uniformly over the arc length (homotopy or continuation method) or optimality in terms of the coverage of the dominated hypervolume (S-metric gradient). Of course, in any of these cases certain assumptions about the function, such as convexity or continuity, have to hold in order to provide guarantees on the quality of the computed set.

7.1 Computing the Maximal Set of Vectors

This section summarizes the algorithm by Kung, Luccio, and Preparata [54] for finding the maxima of a set of vectors with respect to the Pareto order. We use here maximization of the objectives, in order to stay close to the orginal article.

Let S be a subset of $\mathbb{R}^d$. Recall that we have defined a partial order on $\mathbb{R}^d$: for any $v, w \in \mathbb{R}^d$, $v \prec w$ iff for all $i = 1, \cdots, d$,$v_i \leq w_i$ and $\exists j \in \{1, \cdots, d\}$ such that $v_j < w_j$. This partial order is inherited by the subset S.

For $S \subseteq \mathbb{R}^d$ the *maximal set* $\mathrm{Max}(S)$ will be defined as

$$\mathrm{Max}(S) = \{v \in S | \nexists u \in S : v \prec u\}. \tag{7.1}$$

For a Pareto optimization problem, with objectives $f_1, \ldots, f_d$ to be maximized, the maximal set of the image set of the search space $\mathbb{S}$ under $\mathbf{f}$ is the Pareto Front

(PF) of that problem.

For all $d \geq 1$ a lower and upper bound of finding the maximal subset of a set will be given. For the proposed algorithms the time complexity will be derived as well. For $d = 2$ or 3 we will describe efficient algorithms and in that case also prove their efficiency.

The number of comparisons an algorithm A needs for computing the maximal set of S will be denoted with $c_d(A, S)$. The time complexity of a d-dimensional problem in terms of $n = |S|$ is estimated as:

$$\mathcal{T}_d(n) = \min_{A \in \mathcal{A}} \max_{S \subset_n \mathbb{R}^d} c_d(A, S) \tag{7.2}$$

Here $\mathcal{A}$ is the set of all algorithms and $\subset_n$ is the subset operator with the restriction that only subsets of size n are considered.

For $\mathbf{y} \in \mathbb{R}^d$ we denote the vector $(y_2, \cdots, y_d)$ by $\mathbf{y}^*$. In other words, the first coordinate is discarded.

We denote by p_i the projection of S to the i-th coordinate (for any fixed $i \in \{1, \cdots, d\}$).

Definition 79 *Let $A \subseteq \mathbb{R}^d$ and $\mathbf{y} \in S$. Then $\mathbf{y} \prec A$ iff $\exists a \in A$ such that $\mathbf{y} \prec a$*

Lemma 80 *Let $A \subseteq \mathbb{R}^d$ and $\mathbf{y} \in S$. Then $\mathbf{y} \prec A$ iff $\mathbf{y} \prec Max(A)$.* $\square$

A prototype of the algorithm for computing the maximal elements (PF) of S is as follows. In order to present the ideas more clearly we assume $\mathbf{y}^{(i)}, i = 1, \cdots, m$ *to be mutually different in each coordinate.* We shall address equality in some (or all coordinates) separately. It is clear that in case $d = 2$, the sets T_i contain one element and for updates only one comparison is used, so the time complexity of the algorithm in this case is $n \log n$. (The sorting requires $n \log n$ steps while the loop requires n comparisons.)
In case $d = 3$, the maintenance of the sets T_i is done by a balanced binary tree with keys the second coordinate of the vectors $\mathbf{y}^{(i)}$. A crucial step in the loop is to update the T^i given $\mathbf{y}^{(i+1)}$. First determine the element in $T^{(i)}$ that precedes $\mathbf{y}^i$ in its second coordinate. Then determine the elements that are dominated by $\mathbf{y}^{(i)}$ by visiting the leaves of the search tree in descending order of the second coordinates, and stop when the first visited element exceeds the third coordinate of $\mathbf{y}^{(i+1)}$ in its own third coordinate. Discard all visited points, as they are dominated in all three coordinates by $\mathbf{y}^{i+1}$.

Note that in this case the third coordinate is also sorted

By Lemma 80 we can replace the test $\mathbf{y}^{(i)*} \prec T_{i-1}$ in step 5 of Algorithm 3 by the test $\mathbf{y}^{(i)*} \prec \text{Max } (T_{i-1})$.

A variation on Algorithm 3 is to mark an element as maximal as you go along and work with Max (T_{i-1}). Note that the algorithm is 'monotone': once for an

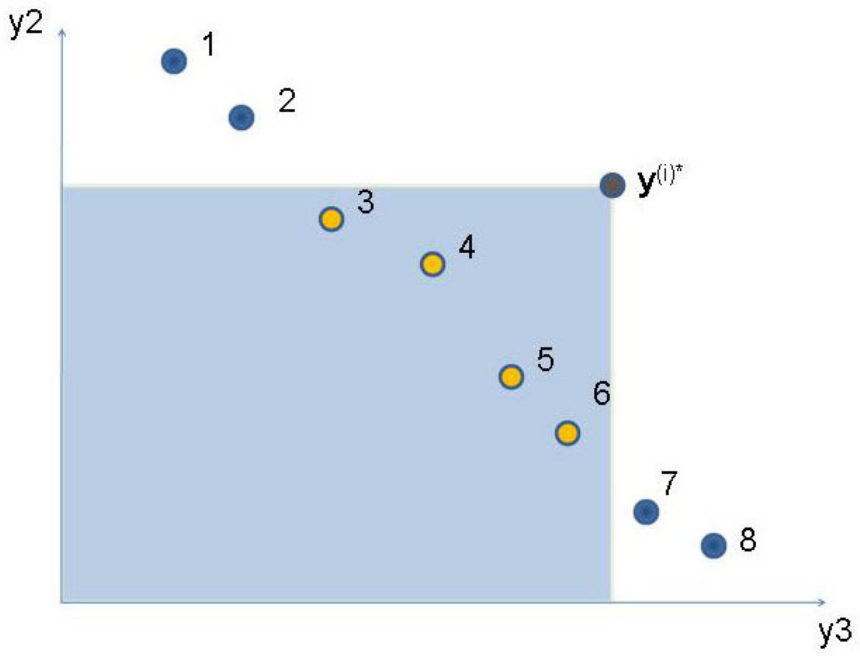

Figure 7.1: In order to find out whether or not a point $\mathbf{y}^{(i)}$ is dominated, first the successor in the 2nd coordinate in ascending order is determined. Using a balanced search tree this step takes $\mathcal{O}(\log n)$ time. If the third coordinate of the successor exceeds the third coordinate of the point $\mathbf{y}^{(i)}$ then $\mathbf{y}^{(i)}$ is dominated and discarded. Otherwise, the algorithm deletes all points that are dominated by $\mathbf{y}^{(i)}$ (points in the shaded area) and insert $\mathbf{y}^{(i)}$ into the non-dominated set and updates the balanced tree. The update of the balanced tree is done by moving through the set of points in direction of decreasing second coordinate (and consequently increasing third coordinate) until the third coordinate of $\mathbf{y}^{(i)}$ is exceeded. The points visited on that path are deleted and finally $\mathbf{y}^{(i)}$ is inserted into the balanced search tree. As each node enters the tree and leaves the tree at most once in the entire n runs of the described algorithm, the total running time will not exceed $\mathcal{O}(n \log n)$.

Algorithm 3 Prototype Algorithm for Computing PF of a *finite* set S

1: **input**: $S = \{\mathbf{y}^{(1)}, \cdots, \mathbf{y}^{(k)}\} \subseteq_k \mathbb{R}^d$ and $k \in \mathbb{N}$
{NB *We assume* the $\mathbf{y}^{(i)}$ *to be mutually different in each coordinate.* }
2: View the elements S as a sequence and sort the elements of this sequence by the first coordinate in descending order: $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \cdots, \mathbf{y}^{(k-1)}, \mathbf{y}^{(k)}$ is now such that

$$p_1(\mathbf{y}^{(1)}) > p_1(\mathbf{y}^{(2)}) > \cdots > p_1(\mathbf{y}^{(k-1)}) > p_1(\mathbf{y}^{(k)})$$

3: $i \leftarrow 1$;
4: $T_0 \leftarrow \emptyset$; { The T_i are sets of $(d-1)$-dim vectors }
5: **for all** $i = 1, \ldots, k$ **do**
6: **if** $\mathbf{y}^{(i)*} \prec T_{i-1}$ **then**
7: $T_i \leftarrow T_{i-1}$
8: **else**
9: $T_i \leftarrow$ Max $(T_{i-1} \cup \{\mathbf{y}^{(i)*}\})$ and mark $\mathbf{y}^{(i)}$ as maximal.
10: **end if**
11: **end for**

element its star is admitted to T_i it is clearly maximal (and the star will survive in future T_i.

The prototype can be specialized by describing which choices have been made for the test $\mathbf{y}^{(i)*} \prec T_{i-1}$ and for the building of the T_i.

Chapter 8

Evolutionary Multicriterion Optimization

Evolutionary Algorithms (*EA*) are stochastic search and machine learning heuristics, inspired by theories of biological evolution, most prominently by the so-called modern evolutionary synthesis that combines natural selection and genetics (see [27]). Traditionally, *EA* have been categorized into three subfields: Genetic Algorithms (*GA*) [37] (mainly binary representations), Evolution Strategies (*ES*) [10] (mainly continuous representations) and Evolutionary Programming (*EP*) [35] (arbitrary representations). Nowadays, the boundaries between these subfields became more fluid and the methods are often grouped together using the term Evolutionary Algorithms (see [3]).

EA are used for machine learning and as simulation models in biology [37], but their major field of application is surely systems optimization. This class of methods does not require derivatives of the functions defining the problem and it is relatively robust and flexible for solving nonlinear optimization problems, due to the stochastic search operators involved in the algorithmic definition. Empirical results suggest that on nonsmooth and nonconvex problems *EA* can outperform classical deterministic methods (see [75]). For continuous optimization, advanced *EA*, such as the Covariance Matrix Adaptation Evolution Strategy (*CMA-ES*) [40], are nowadays considered to be state-of-the-art methods.

Evolutionary Multiobjective Optimization (*EMO*) applies *EA* for the solution of *MOO* problems. After first pioneering work in the eighties [74] and in the early nineties [36, 55], these methods received strong attention during the last two decades [17]. What has emerged as a side-branch of the evolutionary computation community is today considered as a more independent field of research, at the intersection of Evolutionary Algorithms, Metaheuristics, and Multicriteria Decision Making. In addition, *EMO* methods became popular for solving practical problems in multiobjective optimization (see, e.g., Branke et al. [13]). The

major focus in *EMO* has been the computation of finite approximations to Pareto optimal sets and Pareto fronts. Some algorithms, such as the second versions of the Nondominated Sorting Genetic Algorithm (*NSGA-II*) [25] and the Strength Pareto Evolutionary Algorithm (*SPEA2*) [87], became popular techniques for this task and will be reviewed here. We will also review more recent trends, exemplified by algorithms such as the hypervolume-based S-metric Selection *EMO* algorithm (*SMS-EMOA*) [29] and the Multiobjective *CMA-ES* (*MO-CMA-ES*) [46], the latter specially designed for solving continuous optimization problems.

8.1 Single Objective Evolutionary Algorithms

In *EA* mathematical objects are often referred to by biological metaphors, such as individual (solution), population (tuple of solutions), and fitness (objective function value or rank of an individual, within a population).

This adds a layer of abstraction to the algorithmic description, which can present different instantiations. As an example, an individual could represent a point in $\{0,1\}^n$, a point in $\mathbb{R}^n$, or even an instance of a dynamic data-structure. In the sequel, however, the focus will be on continuous search spaces, meaning that individuals are represented as points in $\mathbb{R}^n$ and populations are considered to be fixed-cardinality multisets of individuals.

Algorithm **??** describes a generic *EA* for single objective optimization. Using the subsequently discussed genetic operators, the stochastic transition is designed in such a way that the expected value of the objective function of the population individuals improves over time. Within the framework of *EA* one distinguishes between parent populations X_t (of size $\mu > 0$) and offspring populations O_t (of size $\lambda > 0$), at iterations $t \in \{0, 1, 2, \dots\}$. The iteration counter is denoted by t to emphasize that $\{\mathrm{X}_t\}_{t\in\mathbb{N}}$ could be regarded as the trajectory in $(\mathbb{R}^n)^\mu$ of a stochastic process indexed in time. The offspring population, O_t, is an intermediate population that is generated by the variation operators (recombination and mutation) from the individuals of the parent population X_t. The next generation parent population, X_{t+1}, will be selected from the union of the parent and offspring populations, considering the fitness of the individuals.

The initialization of an *EA* is typically done in an uniform random manner, with respect to the space of feasible solutions Ω. Important instantiations of *mating selection* are:

- **Random selection:** randomly selecting parents, in which case the same parent can be repeatedly chosen. This selection type is typically applied in *ES* [10].

- **Tournament selection:** two (or more) individuals are selected randomly

Generic Evolutionary Algorithm (*EA*)

Initialization

Define all required algorithmic parameters (parent population size (μ), number of recombination partners (ρ), number of offsprings (λ), mutation strength (σ), etc.) and set the generation (iteration) counter $t = 0$. Initialize the parent population, X_0, and assign a fitness to each of its μ individuals, based on the corresponding objective function value (and, possibly, other criteria).

For $t = 0, 1, 2, \ldots$

- **Mating selection:** Select ρ individuals in X_t, that will serve as 'templates' for the individuals in X_{t+1}.
- **Recombination:** Combine the information of the selected individuals (e.g. by means of random crossover or of averaging) in order to create a new population, O_t, of λ offspring individuals.
- **Mutation:** Perturb (some of) the offspring individuals in O_t by means of small random modifications.
- **Fitness assignment:** Evaluate the fitness of each offspring individual, considering the corresponding objective function value (and, possibly, other criteria).
- **Environmental selection:** Select individuals from the offspring population O_t (and, possibly, also from the parent population X_t), in order to form the next generation parent population X_{t+1} (of size μ).

from the parent population and the individual with the best fitness will be kept. This selection type is typically applied in *EP* [35].

- **Roulette wheel selection:** Individuals are selected with a probability or frequency that is proportional to their fitness or rank in the population. This selection type is typically applied in *GA* [37].

Besides the selection of individuals to be recombined and/or mutated, the environmental selection is used to compute the population X_{t+1} from the offspring population O_t, and also possibly from X_t. The $(\mu + \lambda)$-selection and the (μ, λ)-selection strategies are distinguished:

- $(\mu + \lambda)$**-selection**: The best μ solutions in the union $\mathrm{X}_t \cup \mathrm{O}_t$ are selected. Because the size of this union equals $\mu + \lambda$, this selection is termed $(\mu + \lambda)$-selection. For instance, a $(\mu + 2)$-selection denotes a selection of the μ best solutions from the union of a set of μ parents and a set of 2 offsprings.

- (μ, λ)**-selection**: The best μ solutions among the λ offspring solutions, O_t, are selected, meaning that the previous parent population X_t is not considered in the selection. In this case $\mu \leq \lambda$ must hold.

Offspring individuals are generated by means of *recombination* and *mutation* operators. Recombination is used to combine information of two or more parents in order to form an offspring, for instance by averaging or by crossover. Theoretically, it has been shown that for decomposable functions recombination can enhance the frequency of parts of the individuals that contribute to a good fitness [37] and decrease deteriorating effects of mutation [10]. Recombination also serves to create diversity and to exchange information across a population. In the context of *EMO*, it can create solutions that lie in the gaps between existing solutions.

Mutation modifies an individual by a small random perturbation. For instance, a Gaussian mutation of a given individual $x \in \mathbb{R}^n$ corresponds to:

$$x' = x + \sigma z \text{ with } z \sim \mathrm{Normal}(0, \Sigma),$$

where $\mathrm{Normal}(0, \Sigma)$ denotes a multivariate Gaussian distribution with mean 0 and covariance matrix Σ, and σ is a scaling factor. In *ES* it is common to adapt the scaling factor, σ_t, across the different generations. For this, several control schemes were proposed. One of these proposals is based on success-rates (e.g. [75, 10, 48]). In *ES* following a $1/\tau$th-success rule means that σ is multiplied by a constant $0 < \xi < 1$, if the success-rate after a fixed epoch of $c > \tau$ iterations is below $1/\tau$, and divided by ξ otherwise. A value of $\tau = 5$ proved to be a good default setting for a range of problems and is used as a standard setting – hence the name 1/5th

success rule [75]. Alternatively, mutative self-adaptation and derandomized self-adaptation are widely used. Here σ (or even Σ itself) gets part of the individual and undergoes an evolutionary process. For details, see [10] and [40], respectively.

The *CMA-ES* [40] also adapts the covariance matrix Σ during the generations, adjusting the mutation to the local quadratic form of the objective function. For this, it integrates the sample path from the history of the individuals (evolution path). In particular, for continuous objective functions with an high condition number and correlation between the optimization variables, the *CMA-ES* is often preferred to other *ES* instantiations.

The behaviour of almost all evolutionary algorithms can be modelled as an homogeneous Markovian process with kernel $K(\mathrm{X}, A)$, describing the transition probability of a given population $\mathrm{X} \in (\mathbb{R}^n)^\mu$ entering in some set $A \subset (\mathbb{R}^n)^\mu$ in the subsequent generation. For instantiations of transition kernels for *EA* see [71].

Based on this, conditions for complete convergence of *EA* can be established. Complete convergence is a strong type of probabilistic convergence. Lucacs [59] defines it as:

Definition 81 (Complete convergence) *Let Y denote a random variable (which might as well be a constant) and $\{Y_t\}_{t\in\mathbb{N}}$ a sequence of random variables defined on a probability space $(\mathcal{B}, \mathcal{A}, P)$. Then $\{Y_t\}_{t\in\mathbb{N}}$ is said to converge completely to Y, if for any $\epsilon > 0$:*

$$\lim_{t\to\infty} \sum_{i=1}^{t} P\{|Y_i - Y| > \epsilon\} < \infty.$$

Next, let us define $b : \wp((\mathbb{R}^n)^\mu) \to \mathbb{R}$ as the function that assigns to a population in $(\mathbb{R}^n)^\mu$ the objective function value of its best individual. Moreover, for $\epsilon > 0$ and f_* the global minimum of the objective function f (provided it exists), we define $A_\epsilon = \{\mathrm{X} \in (\mathbb{R}^n)^\mu \,|\, b(\mathrm{X}) \leq f_* + \epsilon\}$. Using these definitions, we can state the following result.

Theorem 82 (Corollary 6.3 in [71]) *An* EA *whose Markovian kernel satisfies the condition $K(\mathrm{X}, A_\epsilon) \geq \theta > 0$, for all $\mathrm{X} \in A_\epsilon^c = (\mathbb{R}^n)^\mu \setminus A_\epsilon$, and $K(\mathrm{X}, A_\epsilon) = 1$ for any $\mathrm{X} \in A_\epsilon$, for any $\epsilon > 0$, will converge completely to the global minimum of the real valued function f, regardless of the initial distribution of the individuals.*

It is interesting to note that the condition on the Markovian kernel is satisfied for *ES* using a $(\mu+\lambda)$-selection strategy, considering a positive definite covariance matrix Σ and bounded scaling factors $0 < \sigma_{min} < \sigma < \sigma_{max} < \infty$, provided they are applied to bound constrained problems with a continuous objective function (see [71]).

On simple quadratic functions, Beyer [9] derived expressions showing a linear convergence rate of *ES* with optimally adapted step sizes, using a $(\mu + \lambda)$ or

(μ, λ)-selection strategies, for a dimension $n \to \infty$. For a summary, see also [10]. Empirically, it was shown that most of these results already hold for n of moderate size. Jägersküpper [48] proved that, on quadratic forms, *ES* using a (1+1)-selection combined with a 1/5th success rule for step-size adaptation converge linearly with overwhelming probability, that is, the probability that the distance to the optimum does not halve in $\mathcal{O}(n)$ iterations approaches zero with exponential rate in n.

More than analysing the convergence properties of classical *EA*, proposals were made to change the design of some methods belonging to this class, in order to enforce convergence to a stationarity point. Hart [43] introduced a variant of *ES*, called Evolutionary Pattern Search, which generalizes the Stochastic Pattern Search method proposed in [42]. The new algorithm includes in the mutation operator a set of search directions which forms a positive basis for $\mathbb{R}^n$. It uses update schemes similar to the ones of *DSM*, but opposed to them considers a population of search points and randomizes some of the procedures.

Recently, Diouane et al. [26] have also proposed a slightly modified version of *CMA-ES*, and other similar *ES* instantiations. For objective functions which are locally Lipschitz continuous, they have proved the convergency for a Clarke stationarity point, independently of the points considered in initialization. The proposed modification consists in strategies to update the step size parameter based on different sufficient decrease conditions.

Besides these theoretical results, we note that in the field of *EA*, systematic experimental research plays an important role in the algorithmic development. See [6], for a treatise.

8.2 Evolutionary Multiobjective Optimization

Shifting the attention from single to multiobjective *EA* means, in the first place, to focus in the selection operator design. While in single objective optimization selection procedures can be directly based on the objective function value and in rankings depending on it, in *MOO*, where there are partially ordered objective function vectors, fitness assignment or ranking is not that straightforward. Three main alternatives can be identified in the *EMO* literature, which were proposed in the given order:

1. **Scalarization-based approaches:** The objective function vectors are aggregated into a single scalar value, using simple weighting or an utility function. In order to compute different solutions, the parameters of the utility function are dynamically or randomly changed. Examples are the Vector Evaluated Genetic Algorithm (*VEGA*) [74] and the Random Objective Selection Evolutionary Algorithm (*ROSEA*) [72], the latter based on an earlier framework proposed by [55] for multiobjective *ES*.

2. **Nondominance and diversity approaches:** Solutions are computed based on dominance and diversity. Often, nondominance is prioritised to diversity to achieve convergence to a set of Pareto optimal solutions. Examples of this approach are the Nondominated Sorting Genetic Algorithm (version 2) *NSGA-II* [25] and the Strength Pareto Evolutionary Algorithm (version 2) *SPEA2* [87].

3. **Indicator-based approaches:** In this case, a selection is performed among sets, favouring the ones which present a better performance indicator. This performance indicator should measure the quality of a set as an approximation to the Pareto front of the problem. Indicator based evolutionary algorithms were suggested as an algorithm class in [86], being the hypervolume indicator often chosen as performance indicator. The Hypervolume Estimation Algorithm *HypE* [4] and the S-Metric Selection Evolutionary Multiobjective Optimization Algorithm *SMS-EMOA* [29] are instantiations of indicator based evolutionary algorithms. The same happens for a common variant of the Multiobjective Covariance Matrix Adaptation Evolution Strategy *MO-CMA-ES* [46, 83].

Aggregation based methods were described in the first pioneering papers on *EMO*. In *VEGA* [74], weights of the objective functions are randomly changed, promoting diversity inside the population and its better coverage of the Pareto front. With the same purpose, *ROSEA* [72] changes randomly the objective function used in the selection. Rudolph [72] analysed the convergence properties of *ROSEA* for a quadratic *MOO* problem, establishing a sublinear convergence rate. The major drawback of these aggregation based approaches relies on their inability to capture concave parts of the Pareto fronts. Thus, they have been widely abandoned in the *EMO* field.

The selection mechanisms incorporated in the *NSGA*, its successor *NSGA-II*, and *SPEA2* are prominent examples of the second selection principle: a combination of nondominance and diversity. Basically, these algorithms differ in the ranking, based on nondominance level, and the subsequent diversity measurement. A more detailed discussion follows in Subsection 8.2.1. Other algorithms that fall into this class are the Pareto Archived Evolution Strategy (PAES) [52] and the ϵ-MOEA [24], which partition the objective function space into grid cells to measure diversity. A stronger focus on variation/sampling operators design gave rise to algorithms such as the Multiobjective Estimation of Distribution Algorithm [12], the Differential Evolutionary Multiobjective Optimization (DEMO) [70] and the Multiobjective Covariance Matrix Adaptation Evolution Strategy (*MO-CMA-ES*) [46, 83].

The *SMS-EMOA* [29] and *HypE* [4] can be seen as instantiations of indicator-based methods, the design of which is directly governed by a performance indica-

tor. Because this design principle is a recent trend in the advancement of *EMO* algorithms, we will devote Subsection 8.2.2 to it. The hypervolume indicator was also proposed as a selection criterion in *MO-CMA-ES* [46, 83]. However, the main contribution present in this method is to generalize the single objective *CMA-ES* to *MOO*. We will discuss it in more detail in Subsection 8.2.3.

Another line of algorithmic designs has focused on interacting models of evolution and co-evolution, inspired by the predator-prey approaches. Here, different objectives are represented by different prey-individuals, that simultaneously perform the selection on different aggregated objective functions (see [56, 39]). Though these bio-inspired algorithms provide flexibility and robustness in practical settings [39], their convergence properties remain widely unexplored.

8.2.1 Algorithms Balancing Dominance and Diversity

NSGA-II [25] is probably the most commonly used *EMO* algorithm, partially due to its straightforward, yet effective, design. *NSGA-II* is a *GA* considering a $(\mu+\mu)$-selection strategy, and binary tournament for mating. Its main innovation, as opposed to single objective optimization algorithms, consists in ranking solutions based on a vector valued fitness function, and thereby at the same time considers domination and diversity. Ranking is done in two steps:

1. **Nondominated sorting:** In this step the union of the parent and offspring populations $Q \in \Omega^{\mu+\mu}$ is partitioned into $Q^{(1)}, \ldots, Q^{(k)}$, where $Q^{(1)}$ is the set of nondominated solutions in Q, and $Q^{(i+1)}$ is recursively defined as the nondominated set of $Q - \cup_{j=1}^{i} Q^{(j)}$, for $i = 1, 2, \ldots$. Nondominated sorting will result in $0 < \kappa \leq |Q|$ nonempty sets $Q^{(1)}, \ldots, Q^{(i)}, \ldots, Q^{(\kappa)}$. The index i of the set $Q^{(i)}$ determines the *nondominance level* of the solutions in $Q^{(i)}$.

2. **Crowding distance sorting:** Each set $Q^{(i)}$, $i \in \{1, \ldots, \kappa\}$, obtained by nondominated sorting, can be further sorted by means of the crowding distance. The crowding distance measures the contribution of a solution to the diversity of a population – the higher the crowding distance the better the solution. For some $q \in Q^{(i)}$ it is given by $crowd(q) = \sum_{d=1}^{n} (\Delta_d^+(q, Q^{(i)}) + \Delta_d^-(q, Q^{(i)}))$, where $\Delta_d^+(q, Q^{(i)})$ denotes the next higher d-th coordinate of a point $q' \in Q^{(i)} - \{q\}$ and $\Delta_d^-(q, Q^{(i)})$ denotes the next lower d-th coordinate of a point $q'' \in Q^{(i)} - \{q\}$.

A crucial design principle of *NSGA-II* is to prioritize on the nondominance level. Only in case of incomparable solutions it favours those that better contribute to diversity. In common implementations of *NSGA-II* either classical genetic operators are used or – in the canonical version for continuous search spaces -

simulated binary crossover in combination with polynomial mutation is considered. For details on these operators we refer to the literature [25].

Although *NSGA-II* is an elitist algorithm, it was shown that it can deteriorate, meaning that after some generations the set of approximate solutions might be strictly worse then the previous population [30]. Nevertheless, for population with larger sizes the effects of deterioration tend to disappear[30]. In fact, in a wide range of benchmarks and application problems, *NSGA-II* was reported to yield good approximations of Pareto fronts, in particular for the 2-D case. A similar algorithm to *NSGA-II* that also enjoys wide popularity is *SPEA2* [87]. Instead of using nondominated sorting for ranking in the first step, it computes the *strength* of individuals, by counting how many other individuals it dominates and by how many individuals it is dominated. A clustering method is then used as a selection criterion among individuals of equal strength, in order to promote diversity.

8.2.2 Indicator-based Algorithms

In the algorithms belonging to this class, selection strategies use performance indicators to measure the quality of different sets, as approximations to the Pareto front of a given problem. To do it, without the knowledge of the actual Pareto front, the hypervolume indicator was suggested, due to its favorable properties[88]. Given a reference point $r \in \mathbb{R}^m$, that is dominated by all the approximations considered for the Pareto front (or at least by the actual Pareto front of the problem), the hypervolume indicator measures the size of the space that is dominated by the set approximating the Pareto front and upper bounded by the considered reference point.

Definition 83 (Hypervolume indicator) *The hypervolume indicator (or* S-metric*, from '*Size of space covered*') for some (approximation) set $A \subset \mathbb{R}^m$ and a reference point $r \in \mathbb{R}^m$ that is dominated by all the points in A is defined as:*

$$HI(A) = Vol\{b \in \mathbb{R}^m | b \leq r \wedge \exists a \in A : a \leq b\} = Vol(\cup_{a \in A}[a, r])$$

Here $Vol(.)$ denotes the Lebesgue measure of a m-dimensional set of points, and $[a, r]$ denotes the interval box with lower corner a and upper corner r.

In 2-D this is simply the covered area, and in 3-D the covered volume (see Figure 8.1 for examples). For reviewing some important properties related to this indicator, lets us introduce a comparison operator between sets.

Definition 84 *Given two nondominated sets A and B, A is better than B, which is represented by $A \prec B$, if*

$$\forall b \in B : \exists a \in A : a \leq b \text{ (componentwise) and } \exists b \in B : \exists a \in A : a \prec b.$$

(Error converting to EPS. Consult Lieuwe)

Figure 8.1: Hypervolume indicator in two dimensions for a set $A = \{a_1, ..., a_4\} \subset \mathbb{R}^2$ (left) and in three dimensions for a set $Y = \{y_1, \ldots, y_5\} \subset \mathbb{R}^3$ (right).

The hypervolume indicator is, up to an isomorphism, the only known indicator that presents the following monotonicity property.

Lemma 85 (Zitzler et al. [88]) *For two nondominated sets A and B with $A \prec B$ and $\forall x \in A \cup B : x \prec r$ for a reference point r it holds that $HI(A) > HI(B)$.*

This property makes sure that strict improvements of approximating sets will result in an increment of the hypervolume. Moreover, the following property can be established for the hypervolume indicator.

Lemma 86 (Zitzler et al. [88]) *Let $\prec$ be defined as in Definition 84, and A and B denote two nondominated sets with $\forall x \in A \cup B : x \prec r$ for a reference point r and $HI(A) > HI(B)$. Then $B \not\prec A$.*

Basically, the lemma states that measuring by means of the hypervolume indicator will never favour an approximation set that is strictly worse than the another one. This implies that search strategies that generate a sequence of approximation sets or populations with monotonously increasing hypervolume will not deteriorate [30].

Indicator-based *EMO* algorithms use indicators that measure the quality of the eventually obtained set to directly guide the selection [86]. The idea is instantiated in the S-Metric Selection Evolutionary Multiobjective Optimization Algorithm (*SMS-EMOA*) [29], which considers a $(\mu + 1)$-selection scheme. Like in *NSGA-II*, the ranking procedure could be subdivided into two steps.

- **Nondominated sorting:** The first step is nondominated sorting (like discussed for *NSGA-II*), giving rise to partitions of equal nondominance level $Q^{(1)}, \ldots, Q^{(\kappa)}$.
- **Hypervolume-based ranking:** The partition with worst nondominance level, namely $Q^{(\kappa)}$, is reduced to a subset of $|Q^{(\kappa)}| - 1$ solutions. The subset that covers the biggest hypervolume survives.

The design of the algorithm makes sure that, regardless of the stochastic realization, the hypervolume of the approximating set will either grow or remain the same. From Lemma 86 it can be concluded that the search cannot deteriorate to

strictly worse Pareto front approximations, as this might be the case in *NSGA-II* and *SPEA2*.

The choice of a $(\mu + 1)$-selection scheme is motivated by the efficiency of the subset selection for this special case. It suffices to delete the solution in $Q^{(\kappa)}$ that presents the smallest contribution to the corresponding hypervolume. The contribution of a point $a \in A \subset \mathbb{R}^m$ to the hypervolume defined by the set A is measured by:

$$\Delta\mathrm{HI}(a, A) = \mathrm{HI}(A) - \mathrm{HI}(A - \{a\}) \tag{8.1}$$

The computation of all contributions can be accomplished in asymptotically optimal time $\mathcal{O}(|A| \log |A|)$, for $m = 2$ and $m = 3$ [31]. For higher dimensions, up to now, only algorithms with polynomial, but superquadratical, running time are known [7]. This limits the applicability of *SMS-EMOA* to problems with only a moderate number of objective functions.

Recently, several results have been obtain on the distribution of points that maximizes the hypervolume indicator. A set is said to be μ-optimal relatively to the hypervolume indicator if it presents the biggest hypervolume among all sets of size μ. It has been shown that, if the size of the actual Pareto front is equal or greater than μ, μ-optimal sets consist of only Pareto optimal points. Moreover, in 2-D, when $\mu \to \infty$, the set of points maximizing the hypervolume indicator will form a dense subset on the Pareto front [2].

Recent studies have shown that in some cases the *SMS-EMOA* is not guaranteed to converge to a μ-optimal set (e.g. [1]). This results from its $(\mu + 1)$-selection scheme. To guarantee convergence to a μ-optimal set, a $(\mu + \lambda)$-selection with $\mu \geq \lambda$ that selects the subset of size μ with maximal hypervolume is both sufficient and necessary. For $m = 2$, an algorithm for subset selection with running time $\mathcal{O}(\mu^3)$ is available [1], but for $m > 2$ only algorithms with exponential running time are known.

In contrast to *SMS-EMOA*, the Hypervolume Estimation Algorithm (*HypE*) [4] considers a general $(\mu + \lambda)$-selection scheme, where λ can be bigger than 1. To accomplish practical feasible running times, instead of using exact computation in the subset selection procedures, it applies Monte Carlo sampling for computations related to the hypervolume indicator and a modified fitness assignment. Instead of only considering contributions of single points to the hypervolume indicator, it also considers joint contributions of two or more points to this indicator. The volume of the subspace that is jointly dominated by k points is assigned in equal proportions to the points that dominate it.

As the computational complexity of the hypervolume indicator grows exponentially with the number of objective functions [14], and Monte Carlo approximation is less affected by dimensionality, *HypE* is considered to be a promising algorithm for *MOO* with an high number of components for the objective function [4].

First convergence results in hypervolume-based optimization were recently obtained. Beume et al. [8] introduced an analysis method that allows to generalize convergence results for single objective *EA* to *EMO* algorithms. Using this method, it has been proved the convergence of *SMS-EMOA* with a $(1 + 1)$-selection, and of a modified version of *SMS-EMOA* with a $(\mu + 1)$-selection, that works with multiple reference points and considers tournament selection based on pairwise comparisons. In particular, Beume et al. [8] extended the convergence results of Rudolph [71] and Jägersküpper [48], discussed in Subsection 8.1, to these two algorithms.

Whereas earlier theoretical research focused on either very simple discrete problems [57] or in algorithmic designs that do not address diversity maintenance [73], the findings of Beume et al. [8] can be regarded as first results towards convergence analysis for common *EMO* algorithms in continuous problem classes of practical relevance. For now, results are still confined to particular, uncommon algorithmic instantiations, and more general convergence results would be desirable.

8.2.3 Multiobjective Evolution Strategy

While the previously discussed *NSGA-II*, *SPEA2*, *SMS-EMOA*, and *HypE* can also be used in discrete search spaces, the *MO-CMA-ES*[46] is an *EMO* method that is especially designed for continuous optimization, since it is intended to generalize the single objective *CMA-ES* to *MOO*. As in *SMS-EMOA*, it uses a ranking strategy which is firstly based on nondominated sorting and considers contributions of the hypervolume indicator (see Definition 83) as a secondary ranking criterion. It establishes a complete ordering on the population by prioritising nondominance level to hypervolume contribution. Consider a set $Q^{(i)}$, for some $i \in \{1, \ldots, \kappa\}$, with more than two solutions of equal dominance level. The solution q with the minimal hypervolume contribution is ranked worst, then the solution with minimal hypervolume contribution is detected from the set $Q^{(i)} - \{q\}$ and assigned the second worst rank, and so forth.

The *CMA-ES* and *MO-CMA-ES* work with a Gaussian mutation operator, for which the corresponding covariance matrix Σ is adapted by means of an elaborated scheme that depends on a evolution path and a success probability. A success is declared if an offspring is selected as one of the parents of the next generation. In case of success the scaling factor σ is decreased, while otherwise it is increased. The path of past evolution steps that led to an offspring is integrated, in order to approximate a conjugate direction that is used to scale and rotate the shape of the distribution represented by the covariance matrix Σ. This way, the mutation distribution is adapted, allowing to increase the progress rate for near-quadratic forms with high condition numbers.

Updates of the covariance matrix require, at least, quadratic time. Despite

this large computational effort, the *MO-CMA-ES* is an interesting alternative to algorithms with a fixed covariance matrix, in cases where the scale of the variables largely differs and search proceeds in directions diagonal to the main coordinate axes.

A state-of-the-art instantiation with a detailed description of the update procedures of the *MO-CMA-ES* is found in [83]. So far, a theoretical convergence theory is not available for *MO-CMA-ES*, though there is empirical evidence of its good performance [46]. The improvement of the *MO-CMA-ES* is still a topic of active research. Interesting developments include the introduction of advanced recombination schemes [82], step-size adaptation schemes [83] and replacement of random mating selection by a selection scheme motivated by reinforcement learning [58].

8.3 Final Remarks

EMO algorithms introduced some interesting concepts, such as indicator-based selection and randomization. Some of these might also prove to be beneficial for design of deterministic *MOO* methods. Finally, hybrids of deterministic MOO methods and *EMO*, as they already exists in single objective optimization, might yield search algorithms that enjoy at the same time the flexibility and robustness of *EMO* algorithms and the deterministic convergence properties of *DSM*.

Chapter 9

Conclusion

This work is an overview on Multicritera Optimization and Decision Analysis. To this end it covered the prevalent trends in the field, including the classical theories based on aggregation and the KKT conditions and more modern set-oriented techniques.

The research field is still very active and recently progress is made in the definition of methods for other types of orders, including cone orders or Lorenz orders. Another trend is to combine the MCDM field with optimization methods in order to better meet the needs of practitioners. In practice it will also be necessary to consider human decision making, which is often irrational or based on subjective criteria.

The search for faster and more reliable algorithms for finding Pareto fronts is also an ongoing endeavour. For instance recently online algorithms are developed for the dynamic update of populations based on, for instance, hypervolume. Such updates can be accomplished much faster than recomputing every iteration from scratch. It has for instance been shown by Hupkens and Emmerich that single generations of the 2-D SMS EMOA can be accomplished in logarithmic time complexity, making them very competitive in terms of computational effort to NSGA-II. Also, more recently set oriented versions of classical numerical algorithms such as gradient descent and Newton's methods were formulated on sets, which exhibit super-linear convergence speed, albeit being less robust than evolutionary methods.

Bibliography

[1] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler. Investigating and exploiting the bias of the weighted hypervolume to articulate user preferences. In *GECCO '09: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pages 563–570. ACM, 2009.

[2] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler. Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point. In *FOGA '09: Proceedings of the 10th ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, pages 87–102. ACM, 2009.

[3] T. Bäck and H. P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolut. Comput.*, 1:1–23, 1993.

[4] J. Bader and E. Zitzler. HypE: An algorithm for fast hypervolume-based many-objective optimization. *Evolut. Comput.*, 19:45–76, 2011.

[5] S. Bandyopadhyay, S. K. Pal, and B. Aruna. Multiobjective GAs, quantative indices, and pattern classification. *IEEE Trans. on Syst., Man, and Cybern. – Part B: Cybernetics*, 34:2088–2099, 2004.

[6] T. Bartz-Beielstein. *Experimental Research in Evolutionary Computation: The New Experimentalism*. Springer, 2006.

[7] N. Beume. S-Metric calculation by considering dominated hypervolume as Klee's measure problem. *Evolut. Comput.*, 17:477–492, 2009.

[8] N. Beume, M. Laumanns, and G. Rudolph. Convergence rates of SMS-EMOA on continuous bi-objective problem classes. In *FOGA '11: Proceedings of the 11th ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, pages 243–251. ACM, 2011.

[9] H. G. Beyer. *The Theory of Evolution Strategies*. Springer, 2001.

[10] H. G. Beyer and H. P. Schwefel. Evolution strategies – A compreehensive introduction. *Natural Computing*, 1:3–52, 2002.

[11] Jozef Bialas and Namakura Shinshu: The Theorem of Weierstrass, Journal of Formalized Mathematics, Volume 7 (Online).

[12] P. A. N. Bosman and D. Thierens. The naive MIDEA: a baseline multi-objective EA. In *EMO 2005: Third International Conference in Evolutionary Multi-Criterion Optimization*, LNCS, pages 428–442, 2005.

[13] J. Branke, K. Deb, K. Miettinen, and R. Slowinski (Eds.). *Multiobjective Optimization: Interactive and Evolutionary Approaches*, volume 5252 of *Lecture Notes in Computer Science*. 2008.

[14] K. Bringmann and T. Friedrich. Approximating the least hypervolume contributor: NP-Hard in general, but fast in practice. In *EMO 2009: Fifth International Conference in Evolutionary Multi-Criterion Optimization*, LNCS, pages 6–20, 2009.

[15] J. Brinkhuis and V. Tikhomirov: Optimization: Insights and Applications, Princeton University Press, NY, 2005.

[16] Rudolf Carnap: Introduction to Symbolic Logic and its Application, Dover Publ., New York, 1958.

[17] C. A. C. Coello. Evolutionary multi-objective optimization: a historical view of the field. *Computational Intelligence Magazine, IEEE*, 1:28–36, 2006.

[18] A. L. Custódio, J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente. Direct multisearch for multiobjective optimization. *SIAM J. Optim.*, 21:1109–1140, 2011.

[19] A. L. Custódio, H. Rocha, and L. N. Vicente. Incorporating minimum Frobenius norm models in direct search. *Comput. Optim. Appl.*, 46:265–278, 2010.

[20] Derringer, G.C. and Suich, D.: Simultaneous optimization of several response values, Journal of Quality Technology 12 (4), pp. 214-219

[21] Daskalakis, C., Karp, R. M., Mossel, E., Riesenfeld, S. J., and Verbin, E. (2011). Sorting and selection in posets. SIAM Journal on Computing, 40(3), 597-622.

[22] B. A. Davey, H.A. Priestley: Introduction to Lattices and Orders (Second Edition), Cambridge University Press, UK, 1990

[23] C. Davis. Theory of positive linear dependence. *Amer. J. Math.*, 76:733–746, 1954.

[24] K. Deb, M. Mohan, and S. Mishra. Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions. *Evolut. Comput.*, 13:501–525, 2005.

[25] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. on Evolut. Comput.*, 6:182–197, 2002.

[26] Y. Diouane, S. Gratton, and L. N. Vicente. Globally convergent evolution strategies and CMA-ES. Technical report, Preprint 12-03, Dept. Mathematics, Univ. Coimbra, 2012.

[27] T. Dobzhansky. *Genetics of the Evolutionary Process.* Columbia Univ. Pr., 1970.

[28] Matthias Ehrgott: Multicriteria Optimization, Springer, 2005

[29] M. Emmerich, N. Beume, and B. Naujoks. An EMO algorithm using the hypervolume measure as selection criterion. In *EMO 2005: Third International Conference in Evolutionary Multi-Criterion Optimization*, LNCS, pages 62–76, 2005.

[30] M. Emmerich, A. H. Deutz, R. Li, and J. W. Kruisselbrink. Getting lost or getting trapped: on the effect of moves to incomparable points in multiobjective hillclimbing. In *GECCO '10: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pages 1963–1966. ACM, 2010.

[31] M. Emmerich and C. M. Fonseca. Computing hypervolume contributions in low dimensions: asymptotically optimal algorithm and complexity results. In *EMO 2011: Sixth International Conference in Evolutionary Multi-Criterion Optimization*, LNCS, pages 121–135, 2011.

[32] E. Fermi and N. Metropolis. Los Alamos unclassified report LS–1492. Technical report, Los Alamos National Laboratory, USA, 1952.

[33] Jeff Erickson. 1995. Lower bounds for linear satisfiability problems. In Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms (SODA '95). Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 388-395.

[34] Steven Finch: Mathematical Constants, Chapter: Transitive Relations, Topologies and Partial Orders, Cambridge University Press, 2003

[35] G. B. Fogel, D. B. Fogel, and L. J. Fogel. Evolutionary programming. *Scholarpedia*, 6:1818, 2011.

[36] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In *Genetic Algorithms: Proceedings of the Fifth International Conference*, pages 416–423. CA: Morgan Kaufmann, 1993.

[37] D. E. Goldberg and J. H. Holland. Genetic algorithms and machine learning. *Machine Learning*, 3:95–99, 1988.

[38] A. Götz and J. Jahn: The Lagrange Multiplier Rule in Set-Valued Optimization: SIAM Journal on Optimization, Volume 10 , Issue 2 (1999) Pages: 331 - 344 Year of Publication: 1999 Kluwer Academic Publishers, Boston, 1999.

[39] C. Grimme, J. Lepping, and A. Papaspyrou. Exploring the behavior of building blocks for multi-objective variation operator design using predator-prey dynamics. In *GECCO '07: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, pages 805–812. ACM, 2007.

[40] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolut. Comput.*, 9:159–195, 2001.

[41] Harrington, J.: The desirability function; Industrial Quality Control 21 (10). pp. 494-498

[42] W. E. Hart. A convergence analysis of unconstrained and bound constrained evolutionary pattern search. *Evolut. Comput.*, 9:1–23, 2001.

[43] W. E. Hart. Evolutionary pattern search algorithms for unconstrained and linearly constrained optimization. In *Evolutionary Programming VII*, pages 303–312. Springer-Berlin, 2001.

[44] R. Hooke and T. A. Jeeves. "Direct search" solution of numerical and statistical problems. *Journal of the ACM*, 8:212–229, 1961.

[45] P. D. Hough, T. G. Kolda, and V. J. Torczon. Asynchronous parallel pattern search for nonlinear optimization. *SIAM J. Sci. Comput.*, 23:134–156, 2001.

[46] C. Igel, N. Hansen, and S. Roth. Covariance matrix adaptation for multiobjective optimization. *Evolut. Comput.*, 15:1–28, 2007.

[47] H. L. Pina J. F. Aguilar Madeira and H. C. Rodrigues. GA topology optimization using random keys for tree encoding of structures. *Structural and Multidisciplinary Optimization*, 7:308–313, 1965.

[48] J. Jägersküpper. How the (1+1) ES using isotropic mutations minimizes positive definite quadratic forms. *Theoretical Computer Science*, 361:38–56, 2006.

[49] J. Jahn. *Introduction to the Theory of Nonlinear Optimization*. Springer-Verlag, 1996.

[50] R.L. Keeney and H. Raiffa: Decisions with multiple objectives: preferences and value tradeoffs, Cambridge University Press, 1993

[51] C. T. Kelley. *Implicit Filtering*. SIAM, 2011.

[52] J. D. Knowles and D. W. Corne. Approximating the nondominated front using the Pareto archived evolution strategy. *Evolut. Comput.*, 8:149–172, 2000.

[53] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Rev.*, 45:385–482, 2003.

[54] H.T. Kung, F. Luccio, and F.P. Preparata: On Finding the Maxima of a Set of Vectors , JACM, Vol. 22, No 4, 1975, pp. 469-476

[55] F. Kursawe. A variant of evolution strategies for vector optimization. In *Proceedings of the 1st Workshop on Parallel Problem Solving from Nature*, PPSN I, pages 193–197. Springer-Verlag, 1991.

[56] M. Laumanns, G. Rudolph, and H. P. Schwefel. A spatial predator-prey approach to multi-objective optimization: a preliminary study. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, PPSN V, pages 241–249. Springer-Verlag, 1998.

[57] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolut. Comput.*, 10:263–282, 2002.

[58] I. Loshchilov, M. Schoenauer, and M. Sebag. Not all parents are equal for MO-CMA-ES. In *EMO 2011: Sixth International Conference in Evolutionary Multi-Criterion Optimization*, LNCS, pages 31–45, 2011.

[59] E. Lucacs. *Stochastic Convergence*. Academic Press, 1975.

[60] J. F. Aguilar Madeira, H. L. Pina, E. Borges Pires, and J. M. Monteiro. Surgical correction of scoliosis: Numerical analysis and optimization of the procedure. *International Journal for Numerical Methods in Biomedical Engineering*, 40:227–240, 2010.

[61] B. H. Margolius: Permutations with Inversions, 4, Journal of Integer Sequences, Article 01.1.4 (Electronic Journal),2001

[62] Miettinen, Kaisa. Nonlinear multiobjective optimization. Vol. 12. Springer, 1999.

[63] J. A. Nelder and R. Mead. A simplex method for function minimization. *Comp. J.*, 7:308–313, 1965.

[64] Jorge Nocedal and Stephen J. Wright: Numerical Optimization (Second Edition), Springer 2007

[65] V.D. Noghin. Relative importance of criteria: a quantitative approach. Journal of Multi-Criteria Decision Analysis, 6(6):355-363,1997

[66] Novak, E., and Ritter, K. (1996). Global optimization using hyperbolic cross points. In State of the Art in global Optimization (pp. 19-33). Springer US.

[67] M. J. D. Powell. Direct search algorithms for optimization calculations. *Acta Numerica*, 7:287–336, 1998.

[68] H. J. Prömel, A. Steger, and A. Taraz, Counting partial orders with a fixed number of comparable pairs, Combin. Probab. Comput. 10 (2001) 159-177;

[69] E. Reehuis, J. Kruisselbrink, A. Deutz, T. Bäck, and M. Emmerich. Multiobjective optimization of water distribution networks using SMS-EMOA. In *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2011)*, pages 269–279. International Center for Numerical Methods in Engineering, 2011.

[70] T. Robic and B. Filipic. DEMO: Differential evolution for multi-objective optimization. In *EMO 2005: Third International Conference in Evolutionary Multi-Criterion Optimization*, LNCS, pages 520–533, 2005.

[71] G. Rudolph. *Convergence Properties of Evolutionary Algorithms*. Verlag Dr. Kovac, 1997.

[72] G. Rudolph. On a multi-objective evolutionary algorithm and its convergence to the Pareto set. In *Proceedings of the 5th IEEE Conference on Evolutionary Computation*, pages 511–516. IEEE Press, 1998.

[73] G. Rudolph and A. Agapie. Convergence properties of some multi-objective evolutionary algorithms. In *Congress on Evolutionary Computation (CEC 2000)*, pages 1010–1016. IEEE Press, 2000.

[74] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the First International Conference on Genetic Algorithms and their Applications*, pages 93–100. Lawrence Erlbaum Associates, 1985.

[75] H. P. Schwefel. *Evolution and Optimum Seeking*. Wiley, 1995.

[76] Stadler, P.F.; Flamm, Ch.: Barrier Trees on Poset-Valued Landscapes, Genet. Prog. Evolv. Mach., 4: 7-20 (2003)

[77] V. Torczon. *Multi-Directional Search: A Direct Search Algorithm for Parallel Machines*. PhD thesis, Department of Mathematical Sciences, Rice University, USA, 1989.

[78] V. Torczon. On the convergence of pattern search algorithms. *SIAM J. Optim.*, 7:1–25, 1997.

[79] A. I. F. Vaz and L. N. Vicente. A particle swarm pattern search method for bound constrained global optimization. *J. Global Optim.*, 39:197–219, 2007.

[80] A. I. F. Vaz and L. N. Vicente. PSwarm: A hybrid solver for linearly constrained global derivative-free optimization. *Optim. Methods Softw.*, 24:669–685, 2009.

[81] L. N. Vicente and A. L. Custódio. Analysis of direct searches for discontinuous functions. *Math. Program.*, on line, 2012.

[82] T. Voß, N. Hansen, and C. Igel. Recombination for learning strategy parameters in the MO-CMA-ES. In *EMO 2009: Fifth International Conference in Evolutionary Multi-Criterion Optimization*, LNCS, pages 155–168, 2009.

[83] T. Voß, N. Hansen, and C. Igel. Improved step size adaptation for the MO-CMA-ES. In *GECCO '10: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pages 487–494. ACM, 2010.

[84] W. Zghal, C. Audet, and G. Savard. A new multi-objective approach for the portfolio selection problem with skewness. In *Advances in Quantitative Analysis of Finance and Accounting*, pages 792–802. Airiti Press, 2011.

[85] X. Zhong, W. Fan, J. Lin, and Z. Zhao. A novel multi-objective compass search. In *IEEE International Conference in Progress in Informatics and Computing*, pages 24–29. IEEE Press, 2010.

[86] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In *Proc. 8th International Conference on Parallel Problem Solving from Nature*, PPSN VIII, pages 832–842, 2004.

[87] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering, 2002.

[88] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Trans. on Evolut. Comput.*, 7:117–132, 2003.