

Multicriteria Optimization and Decision Making

Principles, Algorithms and Case Studies

Michael Emmerich and André Deutz
LIACS Master Course: Autumn/Winter 2006



Contents

1	Introduction	4
1.1	Viewing MOO as a task in system design and analysis	6
1.2	Formal Problem Definitions	8
1.3	Pareto domination and incomparability	11
1.4	Formal Definition of Pareto Dominance	12
2	Theoretical aspects of ordered sets	15
2.1	Axiomatic Definition of Orders	15
2.2	Preorders	16
2.3	Partial orders	18
2.4	Linear orders and anti-chains	19
2.5	Hasse diagrams	19
2.6	Comparing ordered sets	20
2.7	Representing orders as cones	23
3	Pareto optima and efficient points	27
3.1	Search Space vs. Objective Space	27
3.2	Global Pareto Fronts and Efficient Sets	29
3.3	Weak efficiency	30
3.4	Characteristics of Pareto Sets	31
3.5	Optimality conditions based on level sets	32
3.6	Local Pareto Optimality	35
3.7	Barrier Structures	38
3.8	Shapes of Pareto Fronts	41
4	Optimality conditions for differentiable problems	46
4.1	Linear approximations	46
4.2	Unconstrained Optimization	47

4.3	Equality Constraints	48
4.4	Inequality Constraints	53
4.5	Multiple Objectives	55
5	Scalarization Methods	57
5.1	Linear Aggregation	58
5.2	Nonlinear Aggregation	60
5.3	Multi-Attribute Utility Theory	61
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	75
I	Algorithms for Pareto Optimization	77
7	Pareto Front Computing with Deterministic Methods	78
7.1	Continuation methods	78

Preface

Real world decision and optimization problems usually involve conflicting criteria. Ideal solutions are rather the exception than the rule. In this course we will deal with algorithmic methods for solving 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 book 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

Multicriteria optimization and decision making is an exiting field of science. 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 multi-objective 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 eye's view of the field. However, in this book 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 people-centric aspects of decision making and one chapter to the problem of selecting, adapting, and evaluating MOO tools for application problems.

1.1 Viewing MOO 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 this book. 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.

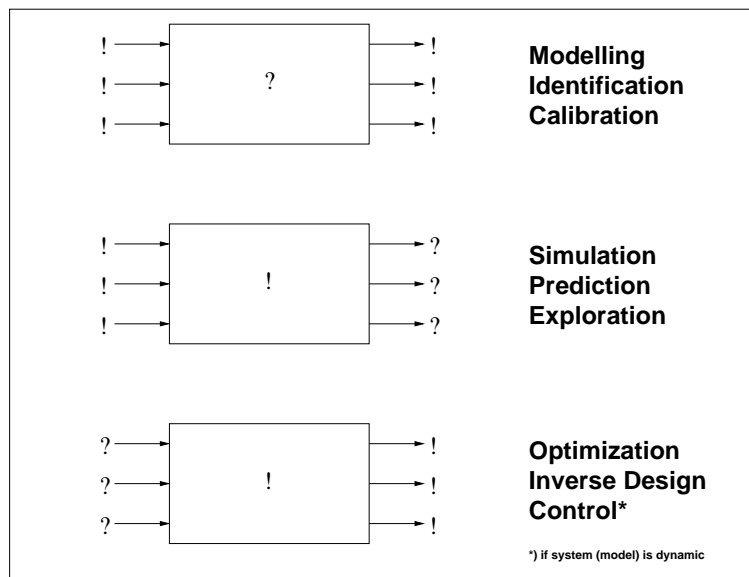


Figure 1.1: Different tasks in systems analysis.

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 this book 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

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

$$f(\mathbf{x}) \rightarrow \min! \quad (* \text{ Objectives } *) \quad (1.1)$$

$$g_1(\mathbf{x}) \leq 0 \quad (* \text{ Inequality constraints } *) \quad (1.2)$$

$$\vdots \quad (1.3)$$

$$g_{n_g}(\mathbf{x}) \leq 0 \quad (1.4)$$

$$h_1(\mathbf{x}) = 0 \quad (* \text{ Equality Constraints } *) \quad (1.5)$$

$$\vdots \quad (1.6)$$

$$h_{n_h}(\mathbf{x}) = 0 \quad (1.7)$$

$$\mathbf{x} \in \mathcal{X} = [\mathbf{x}^{\min}, \mathbf{x}^{\max}] \subset \mathbb{R}^{n_x} \times \mathbb{Z}^{n_z} \quad (* \text{ Box constraints } *) \quad (1.8)$$

$$(1.9)$$

In this definition the objective function f states the main goal of the optimization. It can be evaluated for each search point \mathbf{x} in the search space.

Here the *search space* is defined by a set of intervals, that restrict the range of variables, so called bounds or box constraints.

Whenever inequality and equality constraints are stated explicitly¹, the search space \mathcal{X} can be partitioned in a *feasible search space* $\mathcal{X}_f \subseteq \mathcal{X}$ and *infeasible subspace* $\mathcal{X} - \mathcal{X}_f$. In the feasible subspace all conditions stated in the mathematical program are satisfied. The conditions in the mathematical 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 invariance of mass, an example for an equality constraint). The conditions are called constraints. Due to a convention in the field of operations research, constraints are written in a *standardized form* such that 0 appears on the right side. Equations can easily be transformed into the standard form by means of algebraic operations.

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

Name	Abbreviation	Search Space	Functions
Linear Program	LP	\mathbb{R}^{n_r}	linear
Quadratic Program	QP	\mathbb{R}^{n_r}	quadratic
Integer Linear Program	ILP	\mathbb{Z}^{n_z}	linear
Integer Program	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

Note, that for LP, with the Simplex algorithm there exists a powerful solution technique that, except in rare degenerate cases, solves problems in polynomial running time. For all other problem classes the solution of the general problem is assumed to be intractable. However, in many cases the problems can be solved efficiently if some special structure of the function can be exploited and/or the size of the program or search space is limited. In other cases metaheuristics, such as simulated annealing, evolutionary algorithms or tabu-search, may serve as tools to approximate optimal solutions

¹We do not consider box-constraints as inequality constraints here, as they are usually treated differently by the algorithms.

in practice. We will later give a detailed description of some of these solution methods.

Note that there are also other types of mathematical programs. For instance programs that introduce uncertainties (fuzzy programs, stochastic programs, parametric programs) or programs that are dealing with dynamic data structures, such as parameterized trees and graphs. Moreover, the characteristics of the functions give rise to many definitions of programs, such as semi-definite programs, convex programs etc..

Moreover, in some cases, the framework of mathematical programs is too restrictive. This holds in cases where we optimize complex structures, such as the network topology of recurrent artificial neural networks, or steel joint constructions in bridges. Here the set of solutions can hardly be described as a vector. Rather network models like graph rewriting systems are appropriate to describe the sets of solutions.

In order to capture also these kind of problems a more general definition of a general optimization problem can be used:

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

$\mathbf{x} \in \mathcal{X}$ is called the *search point* or *solution candidate* and \mathcal{X} is the search space or decision space. Finally, $f : \mathcal{X} \rightarrow \mathbb{R}$ 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.

Note, that 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.

In case of multiple objectives the problem definition can be extended to:

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

At this point in time it is not clear, how to deal with situations with conflicting objectives, e.g. when the solutions that minimize f_1 are different from those that minimize f_2 . Note that the problem definition does not yet prescribe how to compare different solutions. To discuss this we will introduce some concepts from the theory of ordered sets, such as the Pareto dominance relation, first.

1.3 Pareto domination and incomparability - An informal example

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 your goal is

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

to minimize the price and maximize speed and you 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 than 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 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.

1.4 Formal Definition of Pareto Dominance

A formal precise definition of Pareto dominance is given as follows.

We define a partial order on the *solution space* $\mathcal{Y} = f(\mathcal{X})$ by means of the Pareto domination 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: $\forall i = 1, \dots, m : y_i^{(1)} \leq y_i^{(2)}$ and $\exists i \in \{1, \dots, m\} : y_i^{(1)} < y_i^{(2)}$.

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 $\mathbf{y}^{(1)} \not\prec_{Pareto} \mathbf{y}^{(2)} \wedge \mathbf{y}^{(1)} \not\preceq_{Pareto} \mathbf{y}^{(2)}$.

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

For any compact subset of \mathbb{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: $\mathcal{Y}_N = \{\mathbf{y} \in \mathcal{Y} | \nexists \mathbf{y}' \in \mathbf{Y} : \mathbf{y}' \prec_{Pareto} \mathbf{y}\}$. 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*.

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 people-centric aspects of these problems?

Chapter 2

Theoretical aspects of ordered sets

The analysis of axiomatic systems describing orders is an essential tool in multi-objective optimization. Next we give a thorough introduction into this topic, showing how orders can be defined as binary relations that satisfy a small number of axioms. Moreover, we will highlight the essential differences between common families of ordered sets, like partial orders, linear orders, and interval 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 Axiomatic Definition of Orders

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 and some common axiomatic properties of binary relations that are relevant for in the context of orders.

A *binary relation* \mathcal{R} on some set \mathcal{S} is defined as a subset of $\mathcal{S} \times \mathcal{S}$. We write $\mathbf{x}^1 \mathcal{R} \mathbf{x}^2 \Leftrightarrow (\mathbf{x}^1, \mathbf{x}^2) \in \mathcal{R}$.

Definition 2.1.1 *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 characteristics 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 $x = y$.

It 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 be 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 2.2.1 *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 2.2.2 *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 2.2.3 *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 2.2.4 *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 $\mathbf{x}^1 \parallel \mathbf{x}^2$.

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:

Minimal elements of a pre-ordered set are elements that are not preceded by any other element.

Definition 2.2.5 *Minimal and maximal elements of an pre-ordered set*

$\mathbf{x}^1 \in \mathcal{S}$ is minimal, iff $\neg \exists \mathbf{x}^2 \in \mathcal{S}$ such that $\mathbf{x}^2 \prec_{pre} \mathbf{x}^1$
 $\mathbf{x}^1 \in \mathcal{S}$ is maximal, iff $\neg \exists \mathbf{x}^2 \in \mathcal{S}$ such that $\mathbf{x}^1 \prec_{pre} \mathbf{x}^2$

For any finite set (except the empty set \emptyset) there exists at least one minimal and 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.

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 2.3.1 *Partial order*

A **partial order** is a preorder that is also *antisymmetric*. We call $(\mathcal{S}, \preceq_{\text{partial}})$ 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 set¹ $\wp(S)$ 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 prePareto 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).

¹the 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 rôle in single criterion optimization, while in the more general case of multiobjective optimization we deal typically with partial orders that are not linear orders. Let us now clarify what essentially distinguishes a linear order from a partial order.

Definition 2.4.1 *Linear order*

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

As we see now, it is only one axiom, the totality 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.

The counterpart of a linear order (also called chain) is the anti-chain.

Definition 2.4.2 *Anti-chain*

A poset $(\mathcal{S}, \preceq_{\text{partial}})$ is said to be an **antichain**, iff: $\forall \mathbf{x}^1, \mathbf{x}^2 \in \mathcal{S} : \mathbf{x}^1 \parallel \mathbf{x}^2$

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. A 'famous' example of a subset of the set of objective function vectors that is an anti-chain is the Pareto front itself.

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 done by so-called Hasse diagrams, named after the mathematician Helmut Hasse (1898 - 1979). The advantage of these diagrams, as compared to the graph representation of binary relations is essentially, that those edges are omitted that can be deduced by transitivity.

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

Definition 2.5.1 *Covers relation*

We say \mathbf{x}^1 is covered by \mathbf{x}^2 , in symbols $\mathbf{x}^1 \triangleleft \mathbf{x}^2$ $:\Leftrightarrow \mathbf{x}^1 \prec_{pre} \mathbf{x}^2$ and $\mathbf{x}^1 \preceq_{pre} \mathbf{x}^3 \prec_{pre} \mathbf{x}^2$ implies $\mathbf{x}^1 = \mathbf{x}^3$.

An equivalent reformulation of the above definition is as follows: \mathbf{x}^2 covers \mathbf{x}^1 iff 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 $\mathbf{x}^1 \triangleleft \mathbf{x}^2$, iff $\mathbf{x}^2 = \mathbf{x}^1 + 1$.

Another example would be the subset relation \subseteq . In this example a set A is covered by a set B if B contains precisely one additional element. In Fig. 2.1 we summarized the subset relation.

A good description of the algorithm to draw a Hasse diagram has been provided by Davey and Priestly [DP90, 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 $\mathbf{x}^1 \triangleleft \mathbf{x}^2$, 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 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, we see 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). We say 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 .

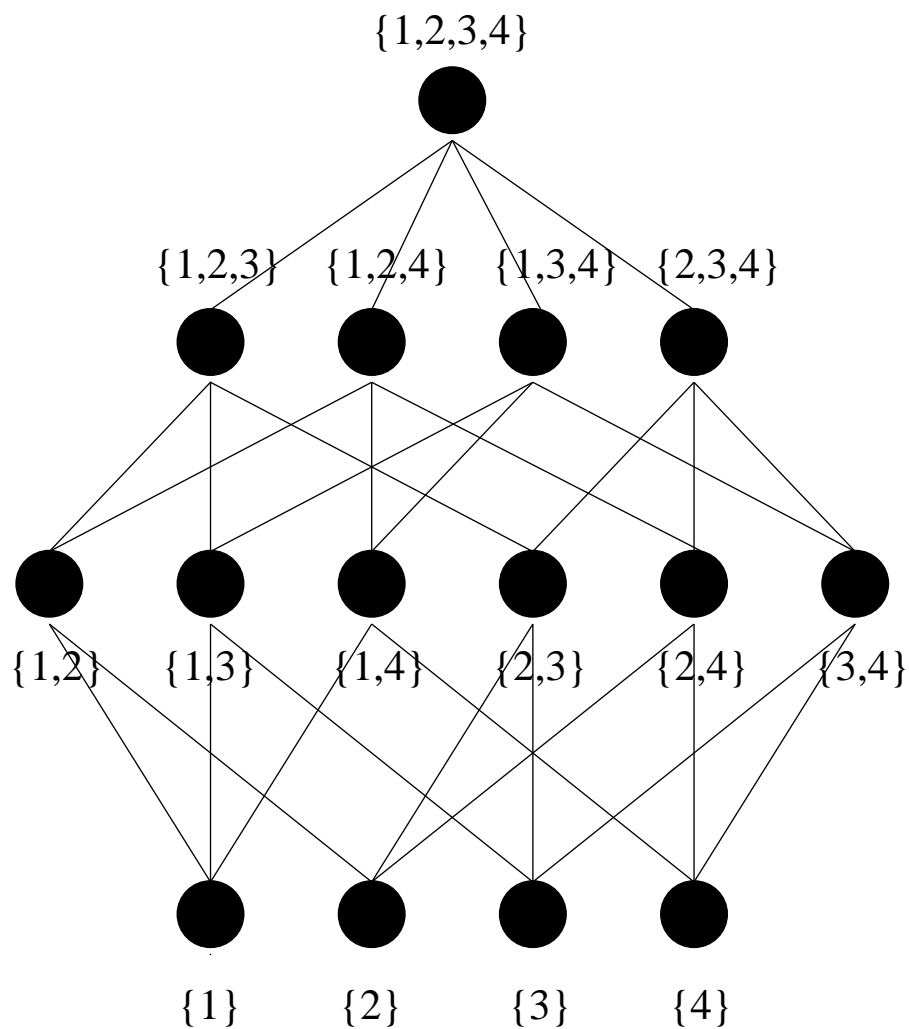


Figure 2.1: The Hasse Diagram for the set of all non-empty subsets partially ordered by means of \subseteq .

Let us now define these concepts formally:

Definition 2.6.1 *An ordered set (X, \preceq) is said to be equal to an ordered set (X', \preceq') , iff $X = X'$ and $\forall x, y \in X : x \preceq y \Leftrightarrow x \preceq' y$.*

Definition 2.6.2 *An ordered set (X', \prec') is said to be isomorphic to an ordered set (X, \preceq) , iff there exists a mapping $\phi : X \rightarrow X'$ 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 2.6.3 *An ordered set (X, \prec') is said to be an extension of an ordered set (X, \prec) , iff $\forall x, x' \in X : x \prec x' \Leftrightarrow x \prec' x'$. 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 2.6.4 *Given two ordered sets (X, \preceq) and (X', \preceq') . A map $\phi : X \rightarrow X'$ 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

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

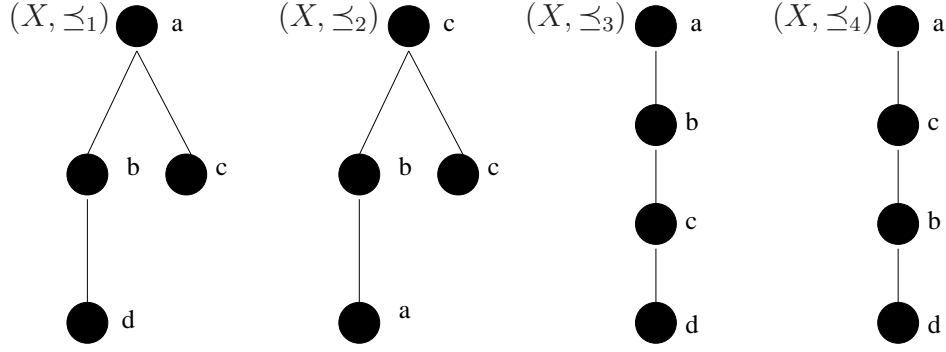


Figure 2.2: Different ordered sets

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

$$\{S_n\}_1^\infty = \{1, 2, 5, 16, 63, 318, 2045, 16999, \dots\} \quad (2.2)$$

. See Finch [6] 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 [6].

2.7 Representing orders as cones

Partial orders in \mathbb{R}^m can be represented as cones. In this section we introduce cones a special types of sets in \mathbb{R}^m . Then we define, how they can be used to represent Pareto orders.

Definition 2.7.1 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 2.7.2 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 2.7.3 *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 2.7.4 *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 $\mathbf{x} \in \mathbf{x}_0 + \mathcal{C}^+$.

Now, let us turn to cones that define (weak, strict) Pareto domination. For this we have to define special convex cones in \mathbb{R} :

Definition 2.7.5 *Orthants*

We define

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

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

Definition 2.7.6 *Pareto domination (defined via cones)*

Given two vectors $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{x}' \in \mathbb{R}^n$:

- $\mathbf{x} < \mathbf{x}'$ (in symbols: \mathbf{x} strictly dominates \mathbf{x}') $\Leftrightarrow \mathbf{x}' \in \mathbf{x} + \mathbb{R}_{>}^n$

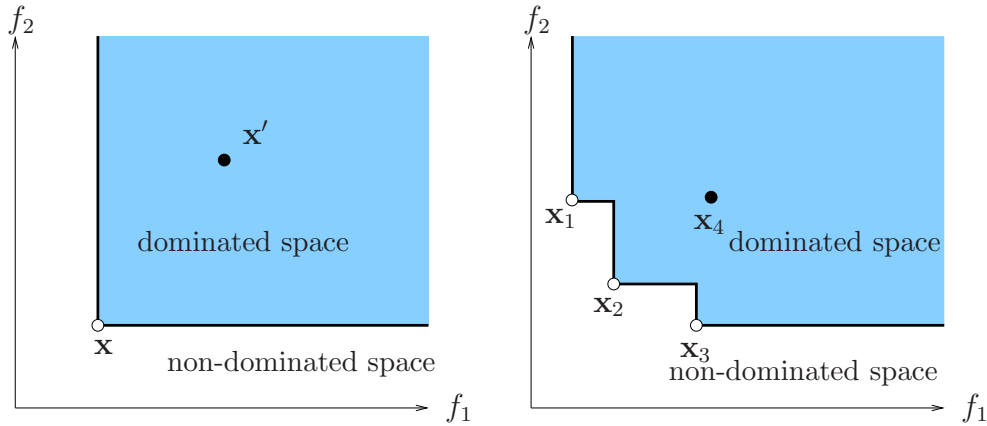


Figure 2.3: Pareto domination in \mathbb{R}^2 defined by means of cones. In the left hand side of the figure the points inside the dominated region are dominated by \mathbf{x} . In the figure on the right side the set of points dominated by the set $A = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$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

- $\mathbf{x} \prec \mathbf{x}'$ (in symbols: \mathbf{x} dominates \mathbf{x}') $\Leftrightarrow \mathbf{x}' \in \mathbf{x} + \mathbb{R}_{<pareto}^n$
- $\mathbf{x} \geq \mathbf{x}'$ (in symbols: \mathbf{x} weakly dominates \mathbf{x}') $\Leftrightarrow \mathbf{x}' \in \mathbf{x} - \mathbb{R}_{\geq}^n$

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 2.7.7 *Domination 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 $\mathbf{x} \in A + \mathbb{R}_{>}^n$, i. e. iff there exists a point $\mathbf{x}' \in A$, such that $\mathbf{x}' \prec_{Pareto} \mathbf{x}$.

Further topics related to cone orders are addressed in [2].

Exercises

1. In definition 2.1.1 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 2.1.1 do they obey!

2. Characterize incomparability (definition 2.2.4) axiomatically! What are the essential differences to indifference?
3. Describe the Pareto order on the set of 3-D hypercube edges $\{(0, 1, 0)^T, (0, 0, 1)^T, (1, 0, 0)^T, (0, 0, 0)^T, (0, 1, 1)^T, (1, 0, 1)^T, (1, 1, 0)^T, (1, 1, 1)^T\}$ by means of the graph of a binary relation and by means of the Hasse diagram!
4. Prove, that $(\mathbb{N} - \{1\}, \preceq)$ with $a \preceq b \Leftrightarrow a \bmod b \equiv 0$ 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

Pareto optima and efficient points

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 than 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 $\mathbf{f} : \mathbb{S} \rightarrow \mathbb{Y}$ 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

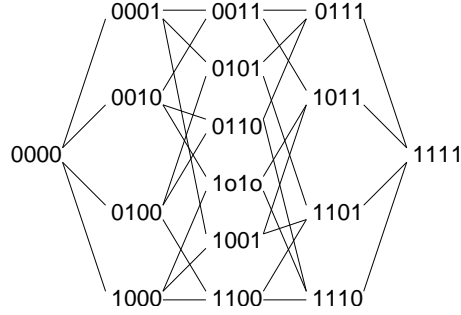


Figure 3.1: The 'binland' example for a discrete partially ordered landscape. The left figure visualizes the Hamming neighborhood on $\{0, 1\}^4$ as adjacency graph.

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 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 3.1.1 *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). $\mathbf{f} : \mathcal{X} \rightarrow \mathbb{R}^m$ is a vector function and \preceq a partial order defined on \mathbb{R}^m . The function $\mathbf{f} : \mathcal{X} \rightarrow \mathbb{R}^m$ will be called height function.*

An example for a poset-valued landscape is given in the Figure 3.8 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

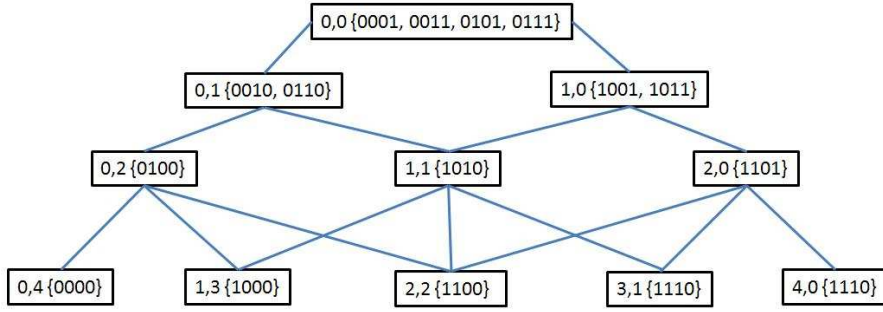


Figure 3.2: Hasse diagram of the Pareto order for the leading ones trailing zeros (LOTZ) problem. The first objective is to maximize the number of leading ones in the bitstring, while the second objective is to maximize the number of trailing zeros. The preorder on $\{0, 4\}^2$ is then defined by the Pareto dominance relation. In this example all local minima are also global minima (compare figure 3.8).

need two essential structures. A neighborhood graph in search space (where edges connect nearest neighbors) the Hasse diagram on the objective space.

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

$$\mathbf{f}(\mathbf{x}) \rightarrow \min, \mathbf{x} \in \mathcal{X} \quad (3.1)$$

Recall that the Pareto front and the efficient set are defined as follows (Section 1.4):

Definition 3.2.1 *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 3.2.2 A point $\mathbf{x}^{(1)} \in \mathcal{X}$ is efficient, iff $\nexists \mathbf{x}^{(2)} \in \mathcal{X} : \mathbf{x}^{(2)} \prec \mathbf{x}^{(1)}$.

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 1.4):

Definition 3.3.1 *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 $\forall i = 1, \dots, m : y_i^{(1)} < y_i^{(2)}$.

Definition 3.3.2 *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} .

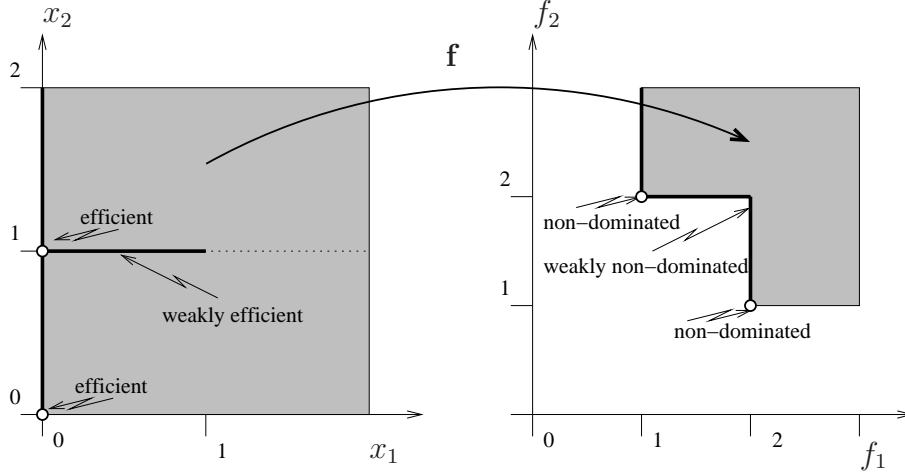


Figure 3.3: Example for a solution set containing weakly efficient solutions.

Example In Fig. 3.3 we graphically represent the efficient and weakly efficient set of the following problem: $f = (f_1, f_2) \rightarrow \min, \mathbb{S} = \mathcal{X} = [0, 2] \times [0, 2]$, 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 3.4.1 *Given an multi-objective optimization problem with m objective functions and image set \mathcal{Y} : The ideal solution is defined as*

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

Accordingly we define the maximal solution:

$$\bar{y} = (\max_{y \in \mathcal{Y}} y_1, \dots, \max_{y \in \mathcal{Y}} y_m).$$

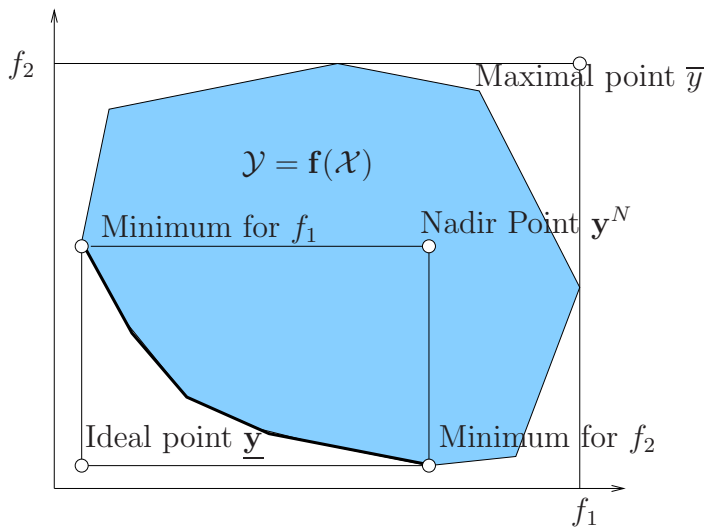


Figure 3.4: Ideal points, Nadir point, and maximal point for a multi-objective optimization problem.

The Nadir point is defined:

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

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 $f : \mathbb{S} \rightarrow \mathbb{R}$, for instance one of the objective functions:

Definition 3.5.1 *Level sets*

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

Definition 3.5.2 *Level curves*

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

Definition 3.5.3 *Strict level set*

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

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

Theorem 3.5.5 The point $\hat{\mathbf{x}}$ can only be weakly efficient if its strict level sets do not intersect. \mathbf{x} is weakly efficient $\Leftrightarrow \bigcap_{k=1}^m \mathcal{L}_<(f_k(\mathbf{x})) = \emptyset$

Theorem 3.5.6 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 $f_1(x_1, x_2) = (x_1 - 1.75)^2 + 4(x_2 - 1)^2 \rightarrow \min$, $f_2(x_1, x_2) = (x_1 - 3)^2 + (x_2 - 1)^2 \rightarrow \min$, $(x_1, x_2)^\top \in \mathbb{R}^2$. 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 $\mathcal{L}_<(f_2(\mathbf{p}_2))$.

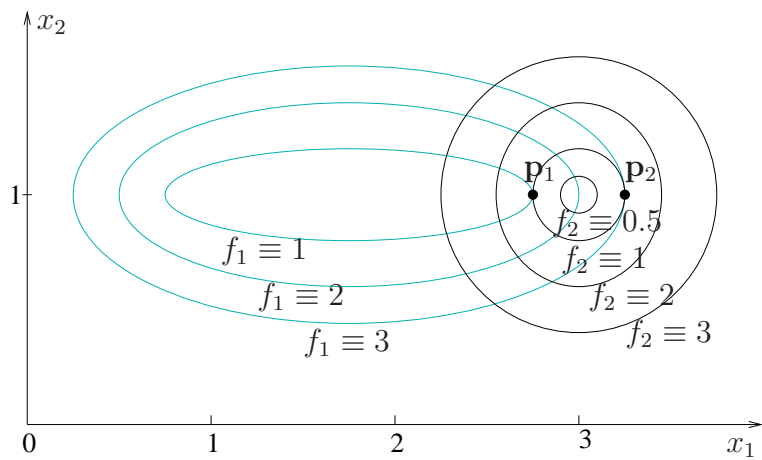


Figure 3.5: Representation of two objective functions as level sets.

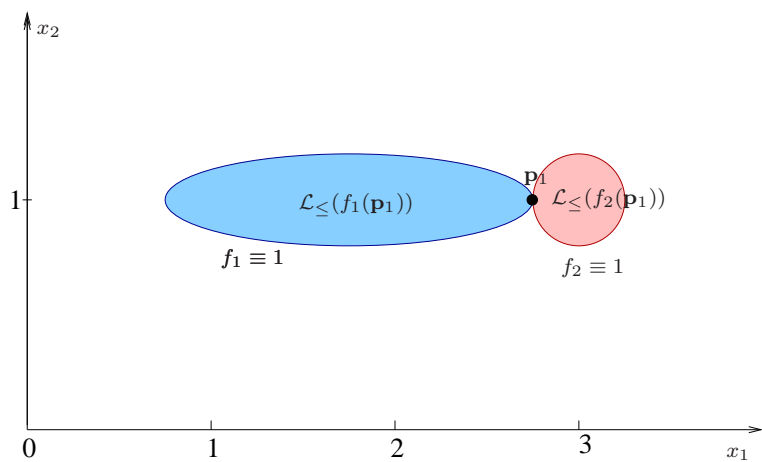
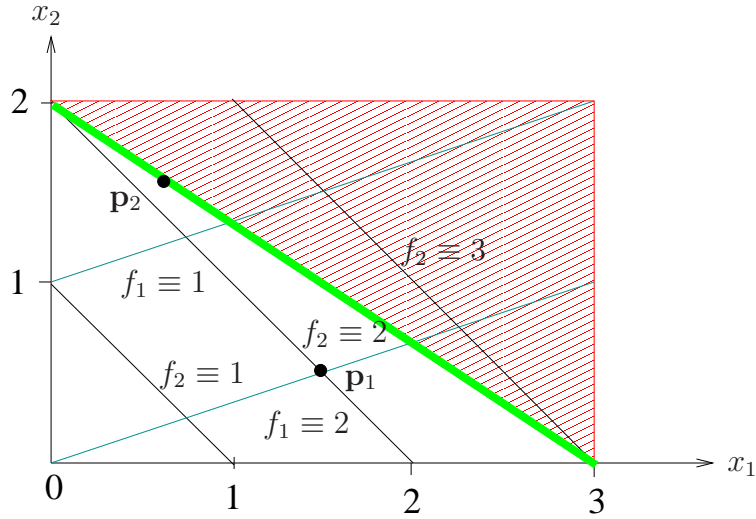


Figure 3.6: The situation for \mathbf{p}_1 . In order to improve f_1 the point \mathbf{p}_1 has to move into the set $\mathcal{L}_{\leq}(f_1(\mathbf{p}_1))$ and in order to improve f_2 it needs to move into $\mathcal{L}_{\leq}(f_2(\mathbf{p}_1))$. Since these sets only meet in \mathbf{p}_1 , it is not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



Example Consider the search space $\mathcal{S} = [0, 2] \times [0, 3]$. Two objectives $f_1(x_1, x_2) = 2 + \frac{1}{3}x_2 - x_1 \rightarrow \min$, $f_2(x_1, x_2) = \frac{1}{2}x_2 + \frac{1}{2}x_1 \rightarrow \max$. 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. Now, we can check different 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 neighborhood function $N : \mathbb{X} \rightarrow \wp(\mathbb{X})$ 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 neighbors of a bit-string x being the elements that differ only in a single bit, i.e. that have a Hamming distance of 1.

Definition 3.6.1 *Locally efficient point (finite search spaces)*

Given a neighborhood function $N : \mathbb{X} \rightarrow \wp(\mathbb{X})$, a locally efficient solution is a point $x \in \mathbb{X}$ such that $\nexists x' \in N(x) : x' \prec x$.

Definition 3.6.2 *Strictly locally efficient point (finite search spaces)*

Given a neighborhood function $N : \mathbb{X} \rightarrow \wp(\mathbb{X})$, 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 3.6.3 *Open ϵ -ball*

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

Definition 3.6.4 *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 3.6.5 *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.

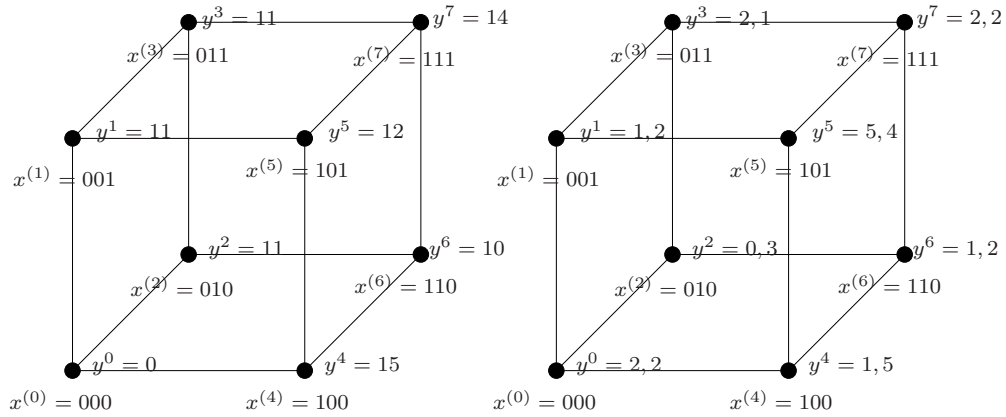


Figure 3.7: Pseudoboolean landscapes with search space $\{0,1\}^3$ and the Hamming neighborhood defined on it. The linearly ordered landscape on the right hand side has three local optima. These are $x^{(0)} = 000, x^{(4)} = 100, x^{(7)} = 111$. $x^{(0)}$ is also a global minimum and $x^{(4)}$ a global 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 $x^{(3)}$

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 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 3.7.1 Path in discrete spaces

Let $N : \mathbb{X} \rightarrow \wp(\mathbb{X})$ be a neighborhood function. A sequence $\mathbf{p}_1, \dots, \mathbf{p}_l$ for some $l \in \mathbb{N}$ and $\mathbf{p}_1, \dots, \mathbf{p}_l \in \mathbb{S}$ 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 $i = 1, \dots, l - 1$, and $\mathbf{p}_l = \mathbf{x}_2$.

Definition 3.7.2 Path in continuous spaces

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

Definition 3.7.3 Let $\mathbb{P}_{\mathbf{x}_1, \mathbf{x}_2}$ denote the set of all paths between \mathbf{x}_1 and \mathbf{x}_2 .

Definition 3.7.4 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 3.8 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} .

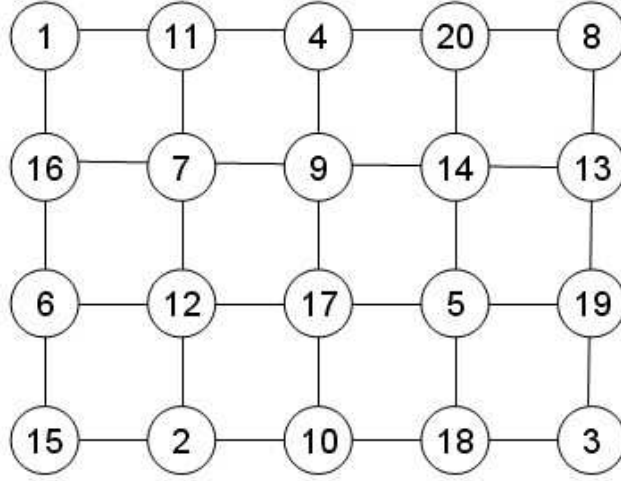


Figure 3.8: Example of a discrete landscape. The height of points is given by the numbers and their neighborhood is expressed by the edges.

Lemma 3.7.5 *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 3.7.6 *The valley (or: basin) below a point \mathbf{s} is called $B(\mathbf{s})$:*

$$B(\mathbf{s}) = \{\mathbf{x} \in \mathbb{S} \mid \exists \mathbf{p} \in \mathbb{P}_{\mathbf{x},\mathbf{s}} : \max_{\mathbf{z} \in \mathbf{p}} f(\mathbf{z}) \leq f(\mathbf{s})\}$$

Example In the aforementioned example given in Figure 3.8, Again, search points x_1, \dots, 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 $\{x_1, x_{11}, x_4, x_9, x_7, x_{13}, x_5, x_8\}$.

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 3.7.7 *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 3.7.7 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 [8] 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 3.7.8 *Maximal points on a path*

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

Definition 3.7.9 *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 optima¹ of $\Sigma_{\mathbf{x}_1, \mathbf{x}_2}$: $S(\mathbf{x}_1, \mathbf{x}_2) := \{\mathbf{z} \in \Sigma_{\mathbf{x}_1, \mathbf{x}_2} \mid \nexists \mathbf{u} \in \Sigma_{\mathbf{x}_1, \mathbf{x}_2} : f(\mathbf{u}) \prec f(\mathbf{z})\}$*

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

¹here we think of minimization of the objectives.

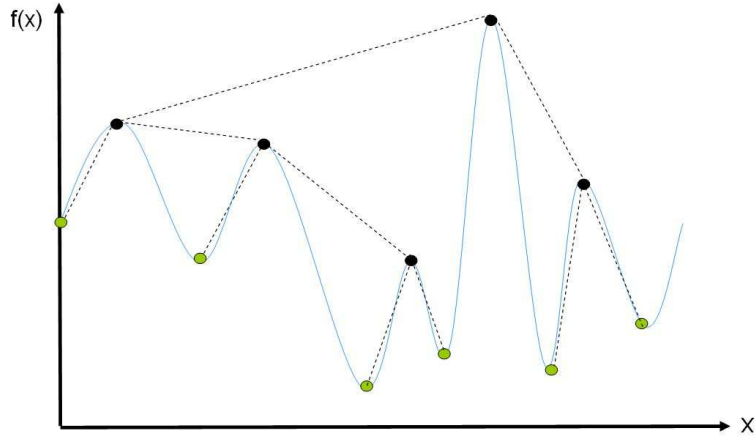


Figure 3.9: A barrier tree of a 1-D continuous function.

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 [8].

A barrier tree for a continuous landscape is drawn in Fig. 3.9. 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.

Definition 3.8.1 *A set $Y \subseteq \mathbb{R}^m$ is said to be cone convex w.r.t. the positive orthant, iff $\mathcal{Y}_N + \mathbb{R}_{\geq}^m$ is a convex set.*

Definition 3.8.2 *A set $Y \subseteq \mathbb{R}^m$ is said to be cone concave w.r.t. the positive*

Algorithm 2 Flooding algorithm

```
1: Let  $x^{(1)}, \dots, x^{(N)}$  denote the elements of the search space sorted in as-
   ascending order.
2:  $i \rightarrow 1$ ;  $\mathcal{B} = \emptyset$ 
3: while  $i \leq N$  do
4:   if  $N(x_i) \cap \{x^{(1)}, \dots, x^{(i-1)}\} = \emptyset$  [i. e.,  $x^{(i)}$  has no neighbour that has
      been processed.] then
5:      $\{x^{(i)}$  is local minimum $\}$ 
6:     Draw  $x^{(i)}$  as a new leaf representing basin  $B(x^{(i)})$  located at the
      height of  $f$  in the 2-D diagram
7:      $\mathcal{B} \leftarrow \mathcal{B} \cup \{B(x^{(i)})\}$       {Update set of basins}
8:   else
9:     Let  $\mathcal{T}(x^{(i)}) = \{B(x^{(i_1)}), \dots, B(x^{(i_N)})\}$  be the set of basins  $B \in \mathcal{B}$ 
      with  $N(x^{(i)}) \cap B \neq \emptyset$ .
10:    if  $|\mathcal{T}(x^{(i)})| = 1$  then
11:       $B(x^{(i_1)}) \leftarrow B(x^{(i_1)}) \cup \{x^{(i)}\}$ 
12:    else
13:       $\{x^{(i)}$  is a saddle point $\}$ 
14:      Draw  $x^{(i)}$  as a new branching point connecting the nodes for
       $B(x^{(i_1)}), \dots, B(x^{(i_N)})$ . Annotate saddle point node with  $B(x^{(i)})$ 
      and locate it at the height of  $f$  in the 2-D diagram
15:      {Update set of basins}
16:       $B(x^{(i)}) = B(x^{(i_1)}) \cup \dots \cup B(x^{(i_N)}) \cup \{x^{(i)}\}$ 
17:      Remove  $B(x^{(i_1)}), \dots, B(x^{(i_N)})$  from  $\mathcal{B}$ 
18:       $\mathcal{B} \leftarrow \mathcal{B} \cup \{B(x^{(i)})\}$ 
19:    end if
20:  end if
21: end while
```

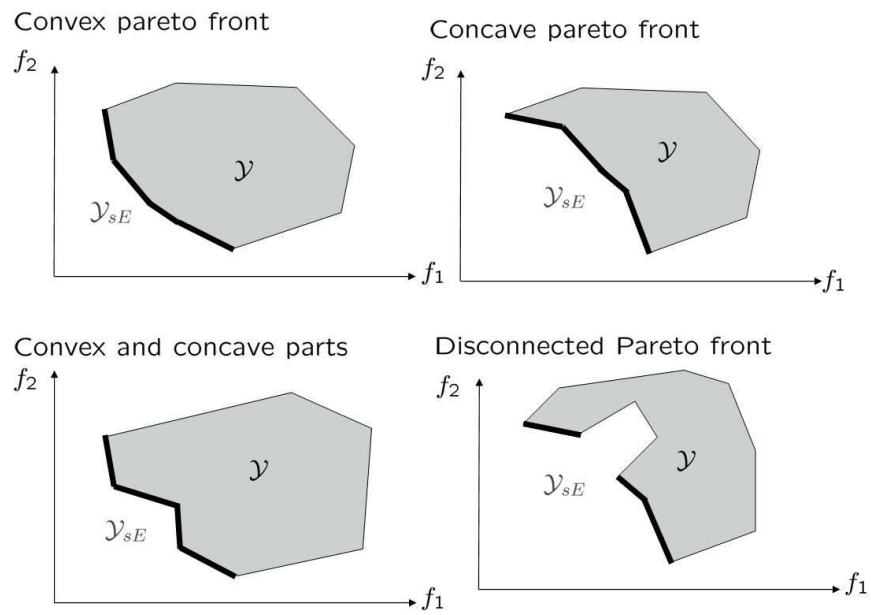


Figure 3.10: Different shapes of Pareto fronts for bi-criteria problems.

orthant, iff $\mathcal{Y}_N - \mathbb{R}_{\geq}^m$ is a convex set.

Definition 3.8.3 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 3.8.4 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 $\phi : [0, 1] \rightarrow \mathcal{Y}_N$ 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. 3.10.

Two further corollaries highlight general characteristics of Pareto-fronts:

Lemma 3.8.5 *Dimension of the Pareto front*

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

Lemma 3.8.6 *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 $\{1, \dots, m\}$ 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

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 $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 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 $\nabla f = (\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n})^\top$. In other words:

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

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.

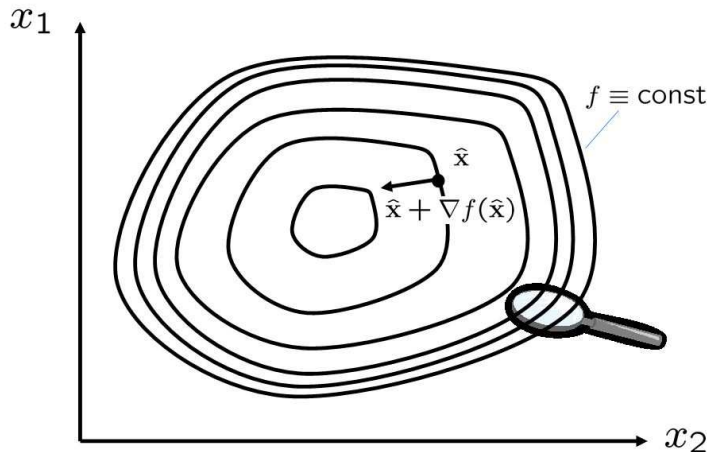


Figure 4.1: Level curves of a continuously differentiable function. Locally the function 'appears' to be a linear function with parallel level curves. The gradient vector $\nabla f(\hat{\mathbf{x}})$ is perpendicular to the local direction of the level curves at $\hat{\mathbf{x}}$.

4.2 Unconstrained Optimization

For the unconstrained minimization

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

problem, a well known result from calculus is:

Theorem 4.2.1 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 whether a matrix is positive (semi)definite:

Theorem 4.2.2 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 [9]. This theorem is also useful for proving the existence of an optimum. This is discussed in detail in [1].

Theorem 4.2.3 *Theorem of Weierstrass*

Let \mathcal{X} be some closed¹ and bounded subset of \mathbb{R}^n , let $f : \mathcal{X} \rightarrow \mathbb{R}$ 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 4.2.1 can be extended to problems with *equality* constraints, i. e.:

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

In this case the following theorem holds:

Theorem 4.3.1 *Let f and g_1, \dots, g_m denote differentiable functions. Then a necessary condition for \mathbf{x}^* to be a local extremum is given, if there exist multipliers $\lambda_1, \dots, \lambda_{m+1}$ with at least one $\lambda_i \neq 0$ for $i = 1, \dots, m$ such that $\lambda_1 \nabla f(\mathbf{x}^*) + \sum_{i=2}^{m+1} \lambda_i \nabla g_i(\mathbf{x}^*) = \mathbf{0}$.*

For a rigorous proof of this theorem we refer to [1]. Let us remark, that the discovery of this theorem by Lagrange preceded its proof by one hundred years [1].

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.

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

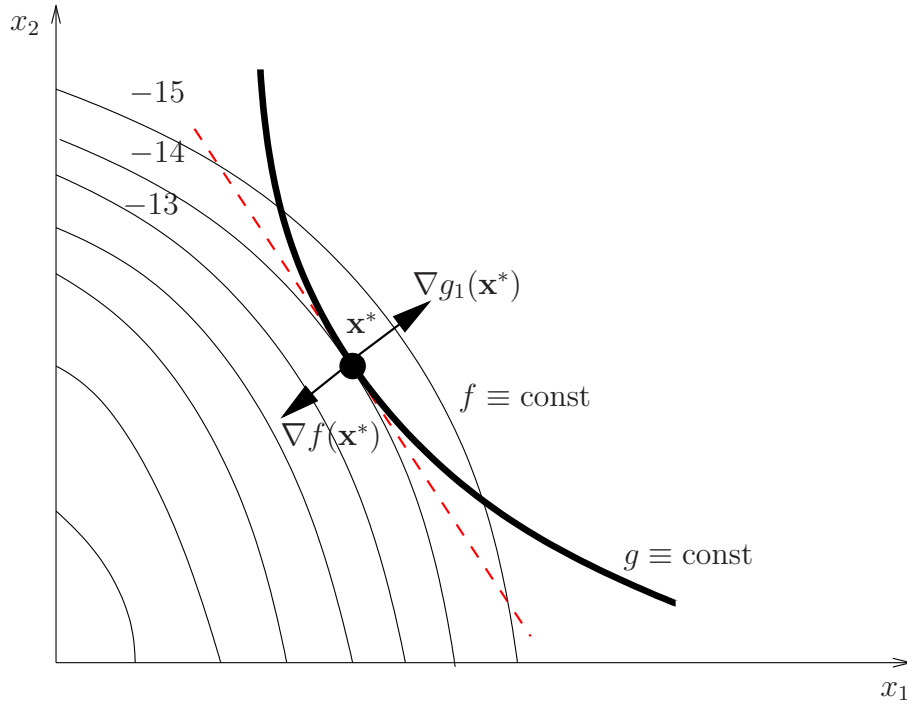


Figure 4.2: Lagrange multipliers: Level-sets for a single objective and single active constraint and search space \mathbb{R}^2 .

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 curves intersect 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 tangential 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 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 $i = 2, \dots, m + 1$. 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:

$$f(x_1, x_2) = x_1^2 + x_2^2 \rightarrow \min \quad (4.4)$$

, with equality constraint

$$g(x_1, x_2) = x_1 + x_2 - 1 = 0 \quad (4.5)$$

Due to the theorem of 4.3.1, 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

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

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

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

$$\lambda_2 = 0 \quad (4.8)$$

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:

$$f|_A \rightarrow \min \quad (4.9)$$

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

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:

$$2x_1 + \lambda_2 = 0 \quad (4.11)$$

$$2x_2 + \lambda_2 = 0 \quad (4.12)$$

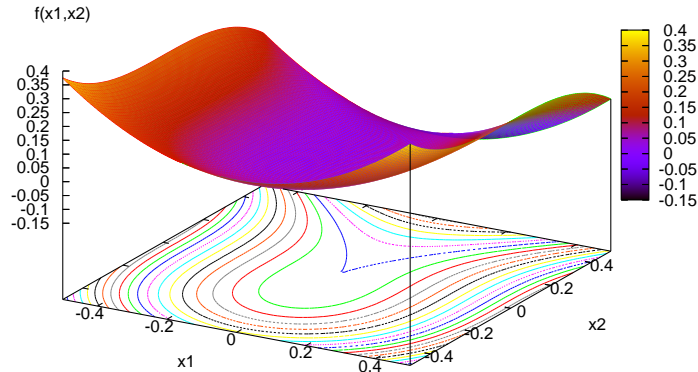


Figure 4.3: The level curves of $x_1^2 - x_2^3$. The level curve through $(0,0)^T$ is cusp.

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 $h(x_2) = (1 - x_2)^2 + x_2^2$. $\frac{\partial h}{\partial x_2} = -2(1 - x_2) + 2x_2 = 0$. 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 [1] give an example of such a 'bad' case:

Example Apply the multiplier rule to $f_0(x) \rightarrow \min, x_1^2 - x_2^3 = 0$: 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.

4.4 Inequality Constraints

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

Theorem 4.4.1 *The Karush Kuhn Tucker conditions are said to hold for \mathbf{x}^* , if there exist multipliers $\lambda_1 \geq 0, \dots, \lambda_{m+1} \geq 0$ and at least one $\lambda_i > 0$ for $i = 1, \dots, m + 1$, such that:*

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

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

Theorem 4.4.2 *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 exist $\lambda_1, \dots, \lambda_{m+1}$ such that KKT conditions are fulfilled.*

Theorem 4.4.3 *The KKT conditions are sufficient for optimality, 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 $\lambda_i = 0$.

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:

$$1 - x^2 \rightarrow \min, x \in [-1, 3]^2 \quad (4.15)$$

First, write the task in its standard form:

$$f(x) = 1 - x^2 \rightarrow \min \quad (4.16)$$

subject to constraints

$$g_1(x) = -x - 1 \leq 0 \quad (4.17)$$

$$g_2(x) = x - 3 \leq 0 \quad (4.18)$$

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

$$\lambda_1 \frac{\partial f}{\partial x} + \lambda_2 \frac{\partial g_1}{\partial x} + \lambda_3 \frac{\partial g_2}{\partial x} = -2\lambda_1 x - \lambda_2 + \lambda_3 = 0 \quad (4.19)$$

$$\lambda_2(-x - 1) = 0 \quad (4.20)$$

$$\lambda_3(x - 3) = 0 \quad (4.21)$$

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:

$$-\lambda_2 + \lambda_3 = 0 \quad (4.22)$$

$$\lambda_2(-x - 1) = 0 \quad (4.23)$$

$$\lambda_3(x - 3) = 0 \quad (4.24)$$

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

$$\lambda(-x - 1) = 0 \quad (4.25)$$

and

$$\lambda(x - 3) = 0 \quad (4.26)$$

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 4.4.3.

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:

$$-2x - \lambda_2 + \lambda_3 = 0 \quad (4.27)$$

$$\lambda_2(-x - 1) = 0 \quad (4.28)$$

, 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 [11]. For multicriterion optimisation the KKT conditions can be generalized as follows:

Theorem 4.5.1 *Fritz John necessary conditions*

A necessary condition for \mathbf{x}^ to be a locally efficient point is that there exist vectors $\lambda_1, \dots, \lambda_k$ and v_1, \dots, v_m such that*

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

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

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

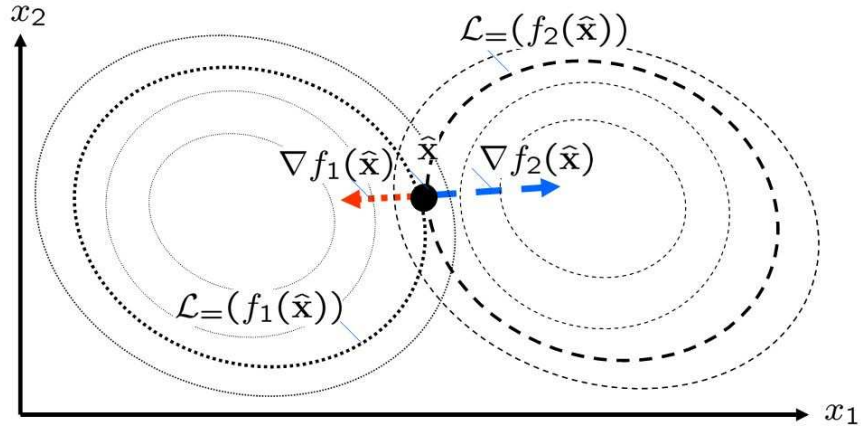


Figure 4.4: Level curves of the two objectives touching in one point indicate locally Pareto optimal points in the bi-criterion case, provided the functions are differentiable.

A sufficient condition for points to be Pareto optima follows:

Theorem 4.5.2 *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 4.5.3 *In the unconstrained case Fritz John necessary 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 (necessary 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 [10].

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:

$$f_1(x) \rightarrow \min, \dots, f_m(x) \rightarrow \min \quad (5.1)$$

is replaced by:

$$\sum_{i=1}^m w_i f_i(x) \rightarrow \min, w_1, \dots, w_m > 0 \quad (5.2)$$

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:

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

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 5.1.1 Proper efficiency [2]

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 $\forall i = 1, \dots, m$ 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.

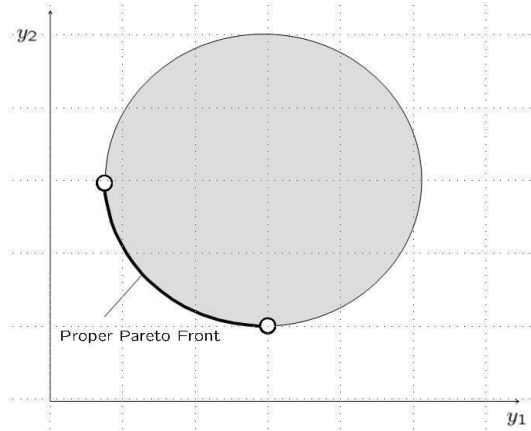


Figure 5.1: The proper Pareto front for a bicriteria problem, for which in addition to many proper Pareto optimal solutions there exist also two non-proper Pareto optimal solutions.

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 5.1.2 *Weighted sum scalarization*

Let us assume a Pareto optimization problem (Eq. 5.1) with a Pareto front that is cone convex w.r.t. positive orthant (\mathbb{R}_{\geq}^m). Then for each properly efficient point $x \in \mathcal{X}$ there exist weights $w_1 > 0, \dots, w_m > 0$ such that x is one of the solutions of $\sum_{i=1}^m f_i(x) \rightarrow \min$.

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 $\sum_{i=1}^m f_i(x) \rightarrow \min$. 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 $w_1y_1 + w_2y_2$ are the solutions obtained with linear

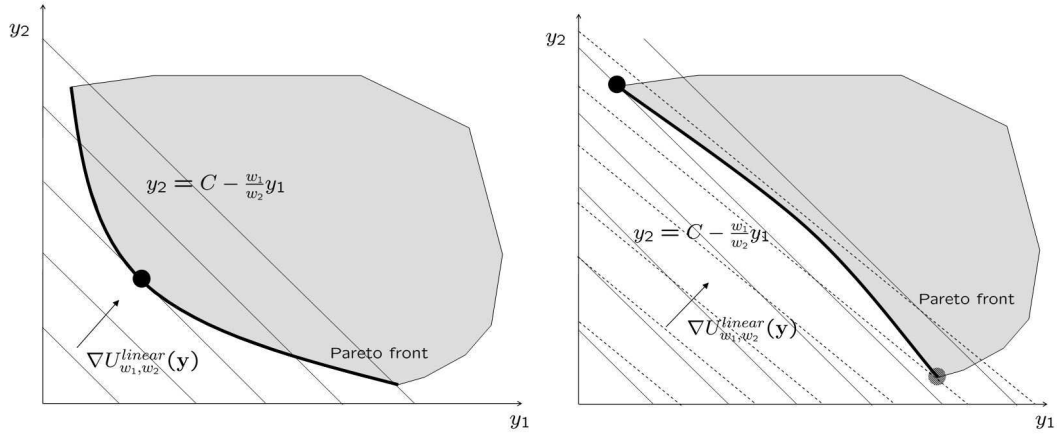


Figure 5.2: The concave (left) and convex Pareto front (right).

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.

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 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 $v_i : \mathbb{R} \rightarrow [0, 10], i = 1, \dots, m$ mapping objective function values to degree of satisfaction values, and their weights $w_i, i = 1, \dots, m$, we can construct the following optimization problem with constraints:

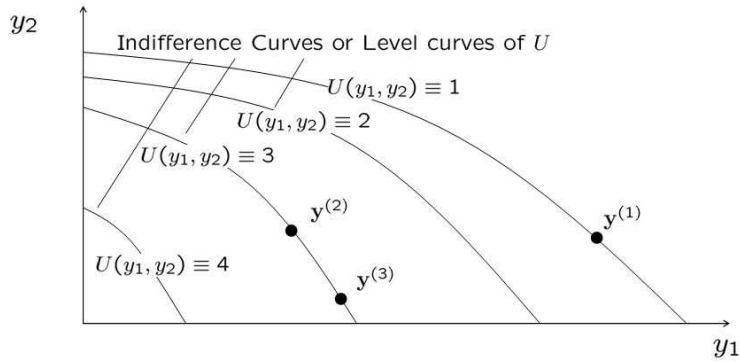


Figure 5.3: Utility function for a bi-criterion problem. If the decision-maker has modeled this utility function in a proper way, he/she will be indifferent whether to choose $\mathbf{y}^{(2)}$ and $\mathbf{y}^{(3)}$, but prefer $\mathbf{y}^{(3)}$ and $\mathbf{y}^{(2)}$ to $\mathbf{y}^{(1)}$.

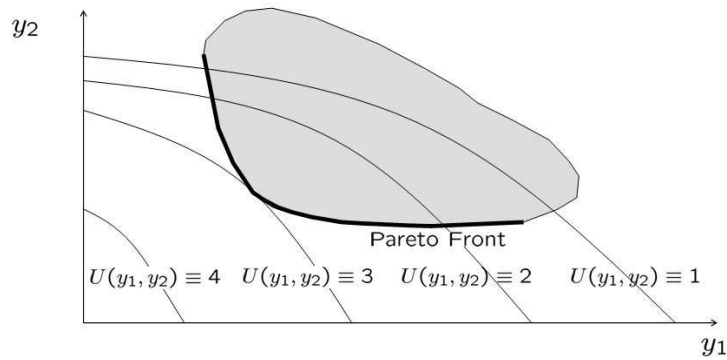


Figure 5.4: The tangential point of the Pareto front with the indifference curves of the utility function U here determines where the solution of the maximization of the utility function lies on the Pareto front.

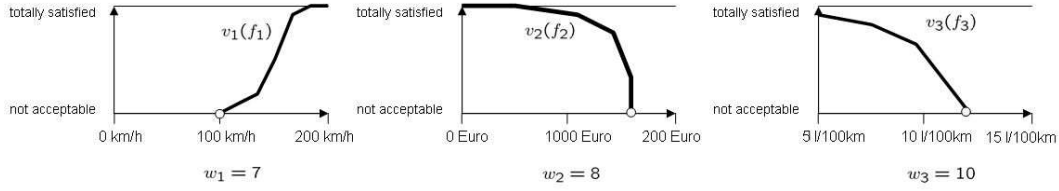


Figure 5.5: The components (value functions) of a multiattribute utility function.

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

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

$$s. t. v_i(f_i(x)) > 0, i = 1, \dots, m \quad (5.6)$$

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 [13] for applications in industrial quality management. Another well known reference for this approach is [14].

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 proposed 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:

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

and the two sided functions read:

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

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

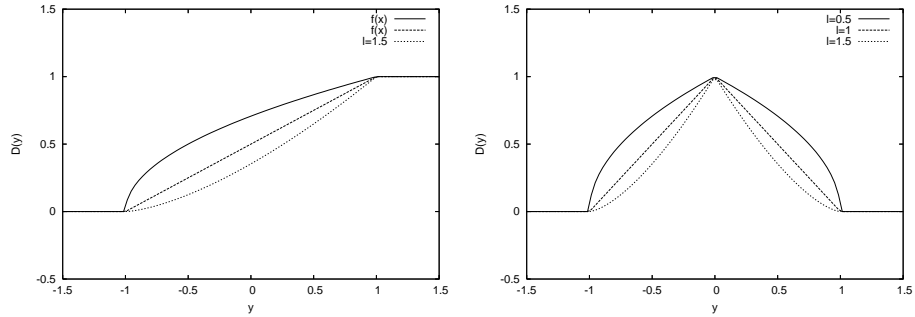


Figure 5.6: In the left figure we see and examples for one-sided desirability function with parameters $y^{min} = -1, y^{max} = 1, l \in \{0.5, 1, 1.5\}$. The left side displays a plot of two sided desirability functions for parameters $y^{min} = -1, y^{max} = 1, l \in \{0.5, 1.0, 1.5\}$, and r being 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:

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

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

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

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

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:

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

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 [12]. Let f_i denote non-negative objective functions:

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

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 :

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

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:

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

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:

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

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:

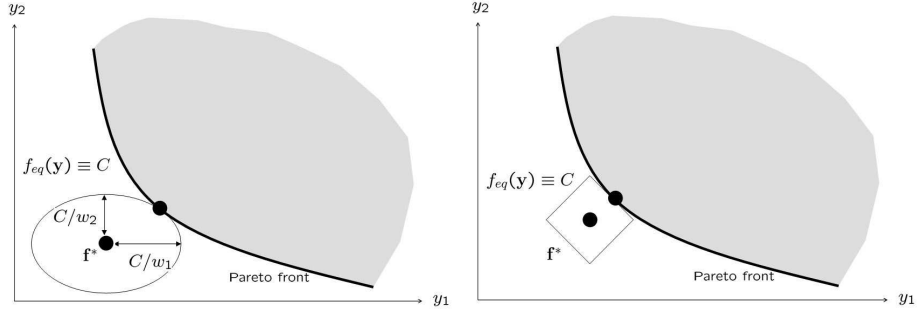


Figure 5.7: Optimal points obtained for two distance to DRP methods, using the weighted Euclidean distance (left) and the Manhattan distance (right).

Lemma 5.4.1 *Let $\mathbf{f}^* \in \mathbb{R}^m$ denote an utopia point, then*

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

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 point 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)

$$d_{\mathbf{w}}^{\infty}(\mathbf{y}, \mathbf{y}') = \max_{i \in \{1, \dots, m\}} w_i |y_i - y'_i| \quad (5.17)$$

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:

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

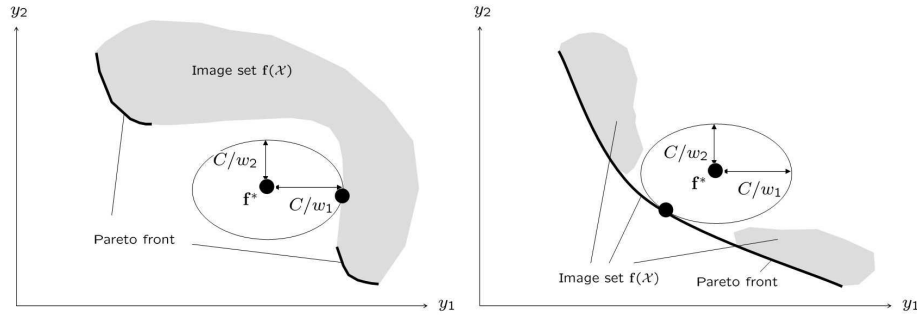


Figure 5.8: In the left figure we see an example for a utopia point which is non-dominated by the image set but the corresponding DRP method does not yield a solution on the Pareto front. In the right figure we see an example where an utopia point is dominated by some points of the image set, but the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

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.

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 to 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.

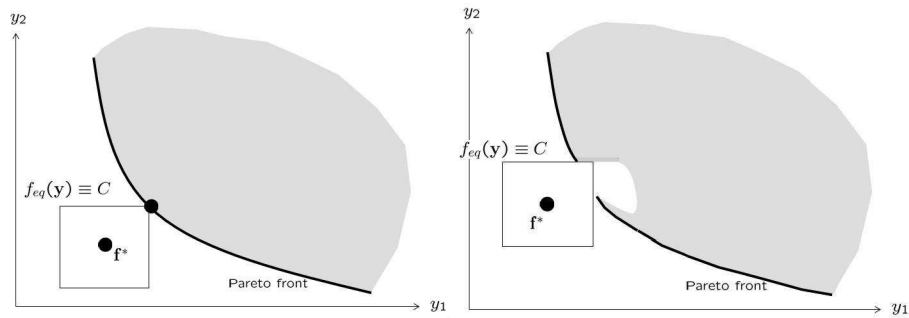


Figure 5.9: In the left figure we see an example where a non-dominated point is obtained using a DRP with the Tschebychev distance. In the right figure we see an example where also dominated solutions minimize the Tschebychev distance to the reference point. In these cases a non-dominated solution may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

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 constraint 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):

$$f_1(x) \rightarrow \min, f_2(x) \rightarrow \min, \dots, f_m(x) \rightarrow \min \quad (6.1)$$

into the single-criterion constrained problem:

$$f_1(x) \rightarrow \min, f_2(x) \leq \epsilon_2, \dots, f_m(x) \rightarrow \epsilon_m. \quad (6.2)$$

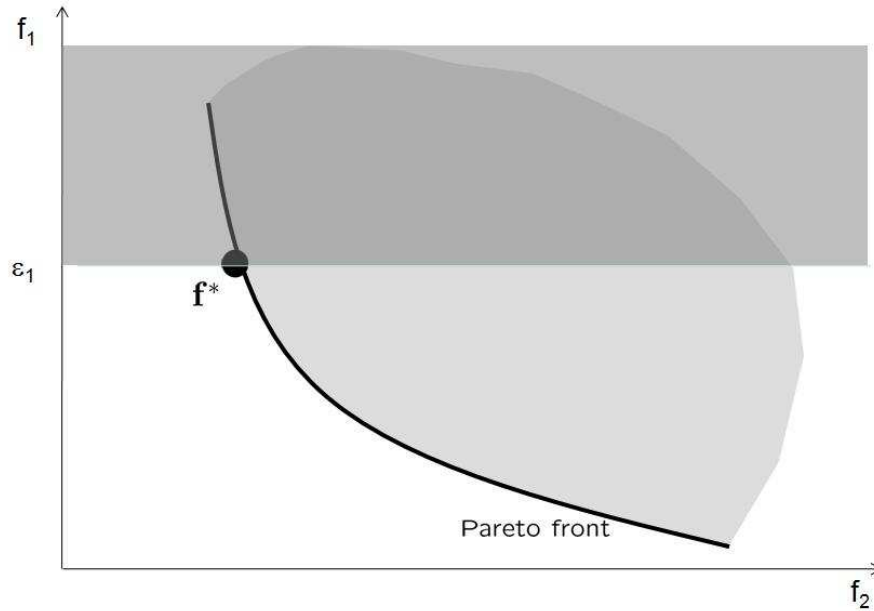


Figure 6.1: Compromise Programming in the bi-criteria case. The second objective is transformed into a constraint.

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, 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 $i = 1, \dots, m$. In these cases, we can obtain optimal solutions using the Lagrange Multiplier method discussed in chapter 4. Not in all cases the solutions we obtain with the compromise programming method are Pareto optimal. An example for a problematic case is given in figure 6.2.

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 3.8.6 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 6.3) we can place points on a grid

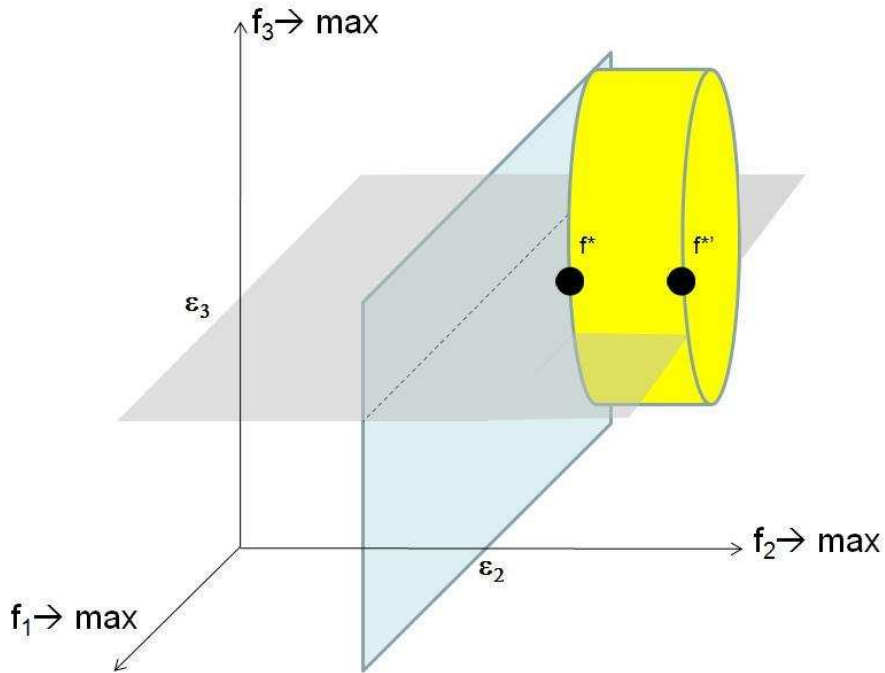


Figure 6.2: Problematic case for the compromise programming method in 3-D. The cheese-like cylinder denotes the image set $f(\mathcal{X})$. The Constraint boundaries are indicated by planes. Note that all objectives are to be maximized. Two of the infinitely many solutions in the image set that qualify as solutions of the constrained problem are indicated by black points. The black point on the right hand side $f^{*'}$ dominates the black point on the left hand side f^* .

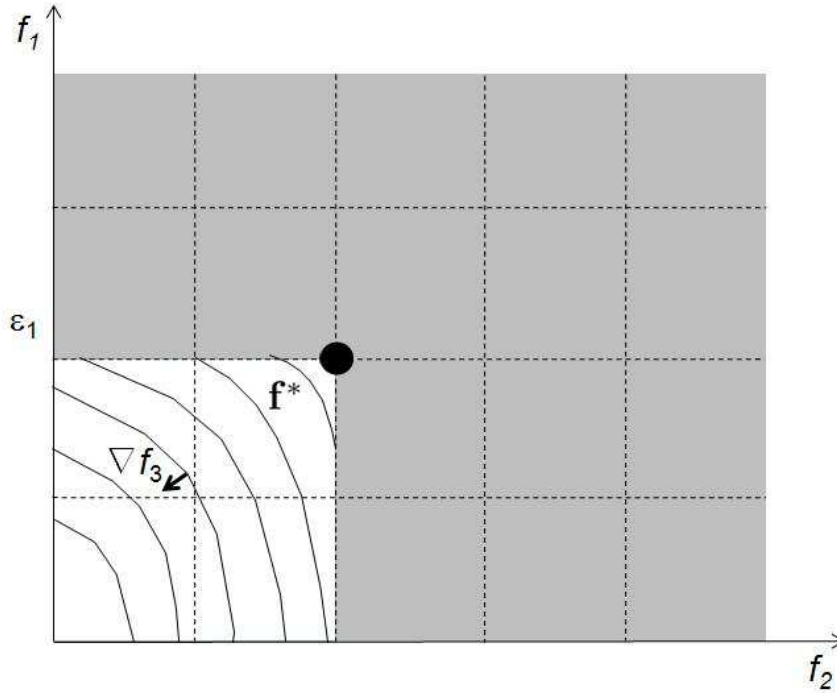


Figure 6.3: Compromised Programming used for approximating the Pareto front with 3 objectives.

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 .

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 6.2.1 *Proper single point method*

A method 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 $A(p, c) = \Omega$.

Definition 6.2.2 *Exhaustive single point method*

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

The following table summarizes the properties of methods we discussed:

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 Tschebyshev 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

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.

Part I

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 Continuation methods

Continuation methods are a class of numerical methods that are used to compute uniformly spaced point sets covering the Pareto front. The basic idea is to start with a single Pareto optimal points and then gradually move to points in its environment and thereby extending the covered surface. The difficulty is to find the right direction and step length in the search space that leads to points in a defined distance of existing points.

If certain conditions are met, systematic ways to generate neighboring points can be derived. In all cases the connectedness of the Pareto front (in the objective space) is assumed. In addition we assume that one connected component of the efficient set will cover the Pareto front.

Next, we will discuss one particular continuation method which will provide all points of a convex 2-D Pareto front.

The idea is to first compute with a single-objective optimization method the two extreme points on the Pareto front, say

$$\mathbf{x}(0) = \arg \min_{\mathbf{x} \in \mathbb{R}^n} f_1(\mathbf{x}) \quad (7.1)$$

and

$$\mathbf{x}(1) = \arg \min_{\mathbf{x} \in \mathbb{R}^n} f_2(\mathbf{x}) \quad (7.2)$$

Next we want to compute a uniformly spaced set of Pareto optimal points on the Pareto front which can be described implicitly as the path $\mathbf{x}(\lambda)$, $\lambda \in [0, 1]$ with

$$\mathbf{x}(\lambda) = \arg \min_{\mathbf{x} \in \mathbb{R}^n} (1 - \lambda)f_1(\mathbf{x}) + \lambda f_2(\mathbf{x}) \quad (7.3)$$

and, assuming convexity and continuous differentiability, this can be expressed as:

$$(1 - \lambda)\nabla f_1(\mathbf{x}) + \lambda\nabla f_2(\mathbf{x}) = 0 \quad (7.4)$$

Note that the extremal points on the Pareto front have as preimages $\mathbf{x}(0)$ and $\mathbf{x}(1)$. The challenge is now to generate a set of points on the Pareto front with a pre-defined distance to each other. To compute this set we start from $\mathbf{x}(0)$ and move successively to neighboring points on the Pareto front, whereby the arclength between neighboring points is approximately given by Δ .

Given a point $\mathbf{x}(\lambda_t)$ we can compute the next point $\mathbf{x}(\lambda_{t+1})$ as follows. First we compute a search direction:

$$\tilde{\mathbf{v}} := \mathbf{x}(\lambda_t + \epsilon) - \mathbf{x}(\lambda_t) \quad (7.5)$$

, where

$$\mathbf{x}(\lambda_t + \epsilon) = \arg \min_{\mathbf{x} \in \mathbb{R}^n} (1 - \lambda_t - \epsilon)f_1(\mathbf{x}) + (\lambda_t + \epsilon)f_2(\mathbf{x}) \quad (7.6)$$

and ϵ is an appropriately small positive number. The normalized $\mathbf{v} := \frac{\tilde{\mathbf{v}}}{\|\tilde{\mathbf{v}}\|}$ is used as the search direction.

Let $F = (f_1, f_2)$. We proceed to compute the step size $h \in \mathbb{R}_+$ along \mathbf{v} in the decision space $\mathbf{x}_{t+1} = \mathbf{x}_t + h\mathbf{v}$ such that $\|F(\mathbf{x}_t) - F(\mathbf{x}_{t+1})\|_\infty = \Theta\Delta$ (where $\Theta \in (0, 1)$ is a safety factor). In case F is Lipschitz continuous we know that there exists an $L \geq 0$ such that

$$\forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \|F(\mathbf{x}) - F(\mathbf{x}')\| \leq L\|\mathbf{x} - \mathbf{x}'\| \quad (7.7)$$

The Lipschitz constant around \mathbf{x}_t can be estimated by:

$$L_{\mathbf{x}_t} := \|DF(\mathbf{x}_t)\|_\infty = \max_{i=1}^2 \|\nabla f_i(\mathbf{x}_t)\|_1 \quad (7.8)$$

Combining 7.7 and 7.8, using $h = \|\mathbf{x}_t - \mathbf{x}_{t+1}\|$, and assuming h is sufficiently small, we obtain the following estimate:

$$h \approx \frac{\Theta\Delta}{L_{\mathbf{x}_t}} \quad (7.9)$$

We can now state how to find the next Pareto optimal point x_{t+1} . Apply the ϵ -constraint method where the constant ϵ is computed as the second coordinate of the expression

$$\epsilon := \pi_2(F(\mathbf{x}_t) + F'(h\mathbf{v})) \quad (7.10)$$

From which we get: $\mathbf{x}_{t+1} := \arg \min f_1(\mathbf{x})$ such that $f_2(\mathbf{x}) = \epsilon$.

Bibliography

- [1] J. Brinkhuis and V. Tikhomirov: Optimization: Insights and Applications, Princeton University Press, NY, 2005.
- [2] Matthias Ehrgott: Multicriteria Optimization, Springer, 2005
- [3] B. A. Davey, H.A. Priestley: Introduction to Lattices and Orders (Second Edition), Cambridge University Press, UK, 1990
- [4] B. H. Margolius: Permutations with Inversions, 4, Journal of Integer Sequences, Article 01.1.4 (Electronic Journal),2001
- [5] H. J. Prömel, A. Steger, and A. Taraz, Counting partial orders with a fixed number of comparable pairs, Combin. Probab. Comput. 10 (2001) 159177;
- [6] Steven Finch: Mathematical Constants, Chapter: Transitive Relations, Topologies and Partial Orders, Cambridge University Press, 2003
- [7] Jorge Nocedal and Stephen J. Wright: Numerical Optimization (Second Edition), Springer 2007
- [8] Stadler, P.F.; Flamm, Ch.: Barrier Trees on Poset-Valued Landscapes, Genet. Prog. Evol. Mach., 4: 7-20 (2003)
- [9] Jozef Bialas and Namakura Shinshu: The Theorem of Weierstrass, Journal of Formalized Mathematics, Volume 7 (Online).
- [10] K. Miettinen: Nonlinear Multiobjective Optimization
- [11] 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.

- [12] R.L. Keeney and H. Raiffa: Decisions with multiple objectives: preferences and value tradeoffs, Cambridge University Press, 1993
- [13] Harrington, J.: The desirability function; Industrial Quality Control 21 (10). pp. 494-498
- [14] Derringer, G.C. and Suich, D.: Simultaneous optimization of several response values, Journal of Quality Technology 12 (4), pp. 214-219