Geometrical Methods in Multivariate Risk Management: Algorithms and Applications

Inauguraldissertation

zur

Erlangung des Doktorgrades

der

Wirtschafts- und Sozialwissenschaftlichen Fakultät

der

Universität zu Köln

2014

vorgelegt

von

Pavel Bazovkin

aus

Simferopol

Referent: Prof. Dr. Karl Mosler

Korreferent: Prof. Dr. Jörg Breitung

Tag der Promotion: 17.11.2014

" . . . I thank Thee, O Father, Lord of Heaven and earth, because Thou hast hid these things from the wise and prudent, and hast revealed them unto babes."

Matt. 11:25

Preface

Starting with the doctoral thesis project, I was impressed by the possibilities that are revealed by geometrical methods in data analysis. First of all, they enable connecting complex methods with a nice interpretation. This, as well as their visuality, was my main motivation for designing this thesis. I have taken an interesting way beginning from generalizing special geometrical algorithms to higher dimensions, then connecting them with concepts from multivariate risk analysis and, finally, ending up with applications to robust optimization. But, of course, I am still on the way.

Acknowledgements

I am deeply grateful to my supervisor, prof. Karl Mosler, who has become for me a real teacher not only in the sphere of his scientific expertise but also in the way of scientific thinking, finding the best ways and in what is called a *scientific style*. He always found a smart way to support me when I needed, and to come into resonance with me when I was inspired. His immense patience and fresh-mindness were important for me.

I am also very thankful to prof. Friedrich Schmid for his support and the thorough consideration of my research. In fact, I was introduced into the sphere of risk management firstly due to his courses and papers. I acknowledge much prof. Jörg Breitung and prof. Roman Liesenfeld for their considering my thesis, support and the cooperation at the Institute and on our research seminars.

I am grateful to my colleagues from Chair of Statistics and Econometrics: prof. Gabriel Frahm, prof. Hans Manner, Daniel Nowak, Walter Orth, Julia Polyakova, Christoph Scheicher and others. A pleasant cooperative atmosphere at the Chair always motivated for a more qualitative research.

I cannot forget my colleagues from the graduate school, many of whom also became my friends: Julius Schnieders, Tobias Wickern, Oliver Grothe, Martin Ruppert, Dominik Liebl and all others. I have learned from them many things, and especially the way of efficient working. If you fill yourself a part of a strong team having a common goal, you get 'wings'. And these 'wings' could hold me only because they were leaning on the fresh air of support and funding provided by the Risk Management section of the Cologne Graduate School (CGS), led by prof. Alexander Kempf. For this, I am grateful to him and other affiliated professors.

Many thanks are coming to the colleagues who were working on similar topics with me - prof. Rainer Dyckerhoff, Pavlo Mozharovskiy and others - for a plenty of fruitful

discussions. Especially to prof. Tatjana Lange, whose vital personality and energy always inspired me to search unconventional solutions for nontrivial problems.

Four of five papers forming this thesis have passed through reviewing process, and the work and critics of almost a dozen of scholars who have thoroughly refereed them was important for improving quality of this thesis. Although not knowing their names, I would like to acknowledge them as well.

Last but not least, I am thankful to my parents, Elena and Evgeny, who were always with me.

Pavel Bazovkin

Cologne, September 2014

Contents

Chapter 1

Introduction

In the recent two decades, the quick development of computational and visualizing instruments led to creating new important scientific directions in multivariate statistical data analysis. In particular, many novel *non-parametric* methods, which are usually computationally demanding, have appeared during this period.

The central theme (leitmotif) of this thesis is also a product of this development. We consider an alternative way of representing a probability distribution which leads, shortly, on the one hand, to rather intuitive instruments and, on the other hand, is freed from any undue parametric modeling. Specifically, we are referring to the *data depth*, which is a function measuring how *deep* a data point is inside a data distribution. It roots in the seminal ideas of John Tukey[1] from the late 1970s. Of course, there are plenty of ways to determine such a function, however, the key fact is that in most cases we need not be bounded by distributional assumptions. This obvious advantage had been impeded for a long time by the usually large complexity of computations. But, beginning from the 1990s, plausible methods and algorithms started to appear in the literature.

Stepping into higher dimensions needed significantly new methods of analysis, which, fortunately, can be found in the sphere of *computational geometry*. The latter developed in parallel starting from rather simple algorithms for convex hulls in dimensions 2 and 3, up to big bundles of modern algorithms such as CGAL[2]. In fact, a possibility of applying geometrical methods with their strong *interpretability* based on the user's intuition was one of the motivations for the opening project of this thesis, especially since many authors started to utilize the connection (cf. Mosler (2004)). This project tackles two problems, (I) and (II), from the list below, where we have collected the aggregate problems that

[1] John Wilder Tukey (1915-2000) was a famous American statistician and author of some novel approaches in statistics and numerical analysis. His paper, Tukey (1975), is widely considered to be the starting point of data depth research.

[2] CGAL, Computational Geometry Algorithms Library, http://www.cgal.org

are considered in the thesis. We will touch on each of them in this Introduction in connection with the thesis' projects incorporated into its chapters.

The list of the aggregate problems:

(I) representing an asymmetric probability distribution in a non-parametric, computationally efficient, and unique way;

⇓

(II) constructing algorithms for so-called central regions (see below) in higher dimensions (> 2);

⇓

(III) measuring multivariate risk comprehensively;

⇓

(IV) comprehensive consideration of risk in portfolio optimization;

⇓

(V) employing the set-valued risk measures in non-financial areas.

According to the depth function, the data can be ordered and in such a manner be represented very intuitively. A further instrument enabling us to visualize the representations is given by the *data trimmed (or central) regions*, which are the sets collecting all points possessing the depth of at least, say, some α. Here α (usually lying in the interval $[0; 1]$) is the parameter of the central region defining its rank inside a family of central regions, thereafter ordered by α in the sense of the inclusion. The most deep region, with $\alpha = 1$, is a median and, correspondingly, included by all other regions.

Such families of trimmed regions also provide a way of representing a probability distribution. Of course, there is a question of whether it is always a one-to-one representation. In a particular simple case of the so-called *Mahalanobis depth*, the *Mahalanobis trimmed regions* are just ellipsoids around the expectation of the probability distribution with a shape coined by the covariance matrix of the distribution. Clearly, in this case the regions can be easily determined in any dimension, but they are defined by just a few parameters (namely, the mean and the covariance matrix). This, obviously, does not allow them to contain the whole information about the distribution.

In this thesis, we are primarily considering such regions whose families describe the corresponding distribution uniquely. For example, the so-called *zonoid trimmed regions* or *halfspace regions* do. That is why it makes sense to be able to compute them. However, it is clear that this problem is much more complex than even the computation of the data depth.

Here we face the aggregate problem (II). It consists in a non-triviality of generalizing computational algorithms from dimension 2 to higher dimensions. The solution of this problem gives us a tool for tackling the further problems (III)-(V), which will be discussed later.

Our first, and simultaneously key contribution is an algorithm that was the first to compute such a, uniquely defining, region in any dimension (Mosler, Lange, and Bazovkin, 2009). Actually, in the next chapter, we propose a solution for the zonoid regions. There, we are also explaining the interpretation of these regions and why they are of high importance. It should be mentioned that the task of computing trimmed regions came into the focus of research in the last decade. However, almost all solutions were limited to data of dimension 2 which, as will be shown in Chapter 2, usually are only qualitatively degenerate cases of the higher dimensions, and, therefore, corresponding algorithms cannot be generalized straightforwardly. The main problem here usually lies in constructing a plausible identification of facets of central regions in dimensions > 2. We solve the problem by proposing the special *spanning tree order* as a general solution of such problems that enables an efficient traversal of all the facets. It replaces the trivial order of facets in dimension 2, which was widely used in the literature. Moreover, our algorithm provides an explicit control of the way how the facets, i.e. the surface of the region, are constructed. It is important when we are interested in building only a part of the region (e.g. its *lower boundary*), or parallelizing the algorithm.

The next step of developing and generalizing the zonoid regions was undertaken by considering the very recently designed notion of the so-called *weighted-mean regions* (Dyckerhoff and Mosler, 2011). For calculating these regions, we have also developed a further algorithm (Bazovkin and Mosler, 2012a), which is a generalization of the algorithm for zonoid regions. The identification of facets is more complex in this general case, however, we are able to employ the similar sequencing principle.

The visuality and the clear interpretation of trimmed regions, as well as the developed mature analytical machinery concerning them, led to the emergence of numerous applications. Basing on some special notions of trimmed regions, recently Cascos and Molchanov (2007) have shown their direct connections with multivariate risk measures, especially set-valued risk measures, which is of a great importance for us. We are contributing to this trend by developing a large class of multivariate set-valued risk measures, the so-called *distortion risk measures*, which possess a list of *desirable properties*. This fact, in turn, motivates their use in a broad variety of applications.

These applications, firstly, cover some classical problems from areas of finance and *risk management*. In particular, we are developing some variants of solving portfolio choice problems with the help of these instruments. Here we are facing problem (IV) because

in the classical model the risk of a portfolio is usually represented by the variance of its random returns. In contrast, we replace the variance by a vector-valued multivariate risk measure based on weighted-mean regions, and show how various *performance measures*, namely objectives, can be applied for the optimization in our approach. Specifically, we propose algorithms for the minimal risk portfolio, a generalized Sharpe ratio and the certainty equivalent. To solve these problems, we have developed a general geometrical framework that is flexible to embedding different performance measures.

Proceeding to problem (V), we should point up that the financial risk management sphere does not limit the applicability of the measures, and we can employ their well-interpretable notions, at a first glance, in a completely different area - the *robust linear optimization*.

We pursue problem (V) from the perspective of modeling uncertainty by means of set-valued risk measures. The uncertainty in the coefficients is modeled by means of special sets, the *uncertainty sets*, which are proved to be special trimmed regions or their simple combinations in the sense of (maybe infinite) disjunction. Each such set is fully defined by a specified risk measure, however, such a visual representation allows us to modify the uncertainty according to additional heuristics, thus flexibilizing the control.

Our contribution in this direction lies in algorithms solving different forms of such optimization problems, ranging from *single-constaint random linear programs* up to rather general forms of *robust convex optimization* (including *robust cone optimization*).

Solving these *stochastic linear programs* (SLPs), from the other side, opens a possibility to apply our algorithms in other important areas, just after finding a way to represent the corresponding problems in the form of SLPs. For instance, we show how a robust data classification problem can be solved using our algorithms.

In more detail, the chapters contribute to the literature in the following way.

The next **Chapter 2** introduces the algorithm for calculating the zonoid trimmed regions. Here, again, we demonstrate that any probability distribution on Euclidean d-space can be described by its zonoid trimmed regions or, in brief, zonoid regions. These regions form a nested family of convex sets – central regions – around the expectation, each being closed and bounded. The family is indexed by numbers that vary in the unit interval. Each zonoid region can be seen as a set-valued parameter that reflects the location, scale, and shape of the distribution.

The motivation for considering such a representation is that the multivariate data are often asymmetrically distributed so that they cannot be modeled by normal or elliptical distributions. Zonoid regions offer a non-parametric and particularly visual approach

to analyzing such data. A distribution - empirical as well as theoretical - is uniquely represented by a geometrical object; its family of zonoid regions. This object is visual and has attractive analytical properties. Moreover, the zonoid regions of an i.i.d. sample satisfy a law of large numbers, converging to the zonoid regions of the underlying probability distribution.

The novelty here consists in:

1. *an analysis* of the representation of a distribution by means of zonoid regions based on their structure and properties;
2. the *spanning tree order* for sequencing facets of a zonoid region;
3. *an exact efficient algorithm* for constructing zonoid regions.

The subsequent **Chapter 3** considers weighted-mean trimmed regions. Starting from theoretical aspects of the regions, we proceed to the main, algorithmic, part where we obtain a generalization of the algorithm for zonoid regions. In doing this, a characterization of a region's facets is used, and information about the adjacency of the facets is extracted from the data. A key problem consists in ordering the facets. It is solved by the introduction of a tree-based order, by which the whole surface can be traversed efficiently with the minimal number of computations. The algorithm has been programmed and is available as an *R* package, which is also described in detail.

The novelty in this chapter splits into:

1. *a special characterization* of facets of a weighted-mean trimmed region;
2. a *tree-based order* for traversing the whole surface of the region;
3. *the generalization* of the algorithm from Chapter 2, a proof of its consistency;
4. an *R* package realizing the algorithm.

In **Chapter 4**, we introduce a *vector-valued multivariate risk measure* that is based on the *set-valued distortion risk measure*. Then, the risk measure is used as a replacement of the *variance* in the classical portfolio choice problem[3]. We build a common geometrical framework, where the portfolio can be optimized according to either the minimal risk, or the Sharpe ratio, or the certainty equivalent.

The novelty of the approach consists in:

[3]Cf., for instance, Markowitz (1952).

1. *a comprehensive* assessment of risk in a portfolio selection problem;
2. taking *no distributional assumptions* concerning ellipticity;
3. *the flexibility* of the framework that enables utilizing various performance measures.

Such an application to the portfolio selection is solely one side of the potential incorporated into the set-valued distortion risk measures. In the further two chapters, we point up a bundle of applications of these risk measures in non-finance environments.

In **Chapter 5**, we apply coherent distortion risk measures to capture the possible violation of a restriction in linear optimization problems whose parameters are uncertain. Each risk constraint induces an uncertainty set of coefficients, which is proved to be a weighted-mean trimmed region. Thus, given a sample of the coefficients, an uncertainty set is a convex polytope that can be exactly calculated. We construct an efficient geometrical algorithm to solve stochastic linear programs that have a single distortion risk constraint. The algorithm's asymptotic behavior is also investigated, when the sample is i.i.d. from a general probability distribution. Finally, we present some computational experience.

The novelty in this chapter splits into three major parts:

1. the uncertainty set of an SLP under a general coherent distortion risk constraint is shown to be a *weighted-mean trimmed region*, which provides a useful visual and computable characterization of the set;
2. an *algorithm* is constructed that solves the minimax problem over the uncertainty set, hence the SLP;
3. proof of the fact that if the data is i.i.d. from a general probability distribution, the uncertainty set and the solution of the SLP are shown to be *consistent estimators* of the uncertainty set and the SLP solution.

In **Chapter 6**, we also investigate linear optimization problems that have random parameters, however, the general case with $m \geq 1$ constraints. In constructing a robust solution $\mathbf{x} \in \mathbb{R}^d$, we control the risk arising from violations of the constraints. This risk is measured by set-valued risk measures, which extend the usual univariate coherent distortion (or, spectral) risk measures to the multivariate case. To obtain a robust solution in d variables, the linear goal function is optimized under the restrictions holding uniformly for all parameters in a d-variate uncertainty set. This set is built from uncertainty sets of the single constraints, each of which is a weighted-mean trimmed

region in $\mathbb{R}^d$ and can be efficiently calculated. Furthermore, a possible substitution of violations between different constraints is investigated by means of the admissable set of the multivariate risk measure. In the case of no substitution, we give an exact geometric algorithm, which possesses a worst-case polynomial complexity.

We extend the algorithm to the general substitutability case, that is, to robust polyhedral optimization. Similarly to the single-constraint algorithm from the previous chapter, the consistency of the approach is proved for generally distributed parameters. Finally, applications of the model, especially applications to supervised machine learning, are discussed.

The novelty of this chapter is contained in:

1. *a generalization* of the analysis of the single-constraint SLP to multiple risk constraints ($m \geq 2$);
2. *a construction* of a geometric algorithm to solve the multi-constraint problem (if constraints cannot be compensated by each other, i.e. in the unsubstitutability case, the algorithm operates in the same dimension d as the single-constraint procedure does);
3. *an extension* of the robust multi-constraint linear optimization to *robust polyhedral optimization* (which covers the substitutability case);
4. *the estimation* (consistent) of the uncertainty set and the robust solution.

Last but not least, in comparison with many recent approaches to determining the uncertainty in coefficients, we are able to work explicitly with uncertainty sets, modifying them if needed.

To summarize, as the reader can see, the chapters are chained in such a manner that the solutions of the above aggregate problems emerge sequentially, in a 'chronological' order. Again, to highlight the unity of the projects as the message of the thesis, we recall the following. *Firstly*, we describe an alternative representation of a probability distribution using central regions, and we construct an exact algorithm for calculating zonoid regions. *Secondly*, keeping in mind the connection of zonoid regions with the set-valued multivariate expected shortfall, we extend our analysis to the broader class of weighted-mean trimmed regions, which, in turn, will correspond to coherent distortion risk measures. We build an exact algorithm for calculating any type of weighted-mean trimmed regions in any dimension. *Thirdly*, based on these regions, we build a set-valued distortion risk measure and, subsequently, a multivariate vector-valued risk measure, which is further

applied in the portfolio optimization. *Fourthly*, we apply the set-valued risk measure in robust convex optimization, obtaining efficient novel optimization algorithms.

The chapters of this thesis are mostly based on the following published papers:

- "Computing zonoid trimmed regions in dimension $d > 2$", with Karl Mosler and Tatjana Lange. *Computational Statistics and Data Analysis*, 53:2500–2510, 2009. Its final version is available at Elsevier via http://dx.doi.org/10.1016/j.csda.2009.01.017.
- "An exact algorithm for weighted-mean trimmed regions in any dimension", with Karl Mosler. *Journal of Statistical Software*, 47(13):1–29, 2012.
- "A geometrical framework for portfolio optimization". *Discussion Papers in Econometrics and Statistics*, Institute of Econometrics and Statistics, University of Cologne, 01/14, 2014.
- "Stochastic linear programming with a distortion risk constraint", with Karl Mosler. *OR Spectrum*, 36(4):949–969, 2014. The final publication is available at Springer via http://dx.doi.org/10.1007/s00291-014-0372-9.
- "A general solution for robust linear programs with distortion risk constraints", with Karl Mosler. *Annals of Operations Research*, 229(1):103–120, 2015. The final publication is available at Springer via http://dx.doi.org/10.1007/s10479-015-1786-8.

Chapter 2

Multivariate Expected Shortfall: Computing Zonoid Trimmed Regions of Dimension $d > 2$

A probability distribution on Euclidean d-space can be described by its zonoid regions. These regions form a nested family of convex sets around the expectation, each being closed and bounded. The zonoid regions of an empirical distribution introduce an ordering of the data that has many applications in multivariate statistical analysis, e.g. cluster analysis, tests for multivariate location and scale. In risk analysis, they can be used as a basis for defining a multivariate expected shortfall risk measure. In this chapter we develop an exact algorithm to constructing the zonoid regions of a d-variate empirical distribution by their facets when $d \geq 3$. We propose a way of characterizing the vertices of the region and their adjacency, and suggest a procedure by which all vertices and facets can be determined. The resulting algorithm has been developed into an R package.

2.1 Motivation

Zonoid trimmed regions or, in brief, zonoid regions, are an alternative way of describing a probability distribution on Euclidean d-space. These regions form a nested family of convex sets – so-called central regions – around the expectation, each being closed and bounded. The family is indexed by numbers that vary in the unit interval. Each zonoid region can be seen as a set-valued parameter that reflects the location, scale, and shape of the distribution.

Multivariate data are often asymmetrically distributed so that they cannot be modeled by normal or elliptical distributions. Zonoid regions offer a non-parametric and particularly visual approach to analyzing such data. A distribution - empirical as well as theoretical - is uniquely represented by a geometrical object; its family of zonoid regions. This object is visual and has attractive analytical properties. Moreover, the zonoid regions of an i.i.d. sample satisfy a law of large numbers, converging to the zonoid regions of the underlying probability distribution.

Zonoid regions were introduced by Koshevoy and Mosler (1997) and, since then, have found many applications in multivariate statistical analysis. They have been employed, e.g., in cluster analysis (Mosler and Hoberg, 2006), in the measurement of inequality (Koshevoy and Mosler, 2007) and polarization (Gigliarano and Mosler, 2009), and in tests for multivariate location and scale (Dyckerhoff, 2002); see also the monograph by Mosler (2002) and a comprehensive introduction to depth statistics by Mosler (2013). Cascos and Molchanov (2007) propose a general geometric framework for measuring multivariate risks; in their approach zonoid regions serve as set-valued risk measures that generalize the usual univariate *expected shortfall*. We will consider this connection in Chapter 4 and employ in the subsequent chapters. In turn, the current chapter prepares necessary tools for computing the measure via computing zonoid regions.

The boundary of a zonoid region forms a depth contour with respect to zonoid depth and can be regarded as a multivariate quantile. Therefore, given a d-variate empirical distribution, zonoid regions are used as trimmed regions that exclude "outlying" data and include "inlying", that is to say, central and relevant ones. Similar methodology has been based on alternative notions of data depth and trimmed regions, such as halfspace (= location) depth, simplicial depth, expected convex hull depth, among others. Zuo and Serfling (2000) provide some general theory of depth trimmed contour regions, while Liu et al. (1999) and Serfling (2006) broadly survey the theory and applications of various notions of depths in multivariate data. López-Pintado and Romo (2007) investigate depth notions in functional data.

When applying such methods to given multivariate data, the crucial point is the availability of efficient numerical procedures to compute the data depths employed and the trimmed regions of an empirical distribution. To calculate the depth of a single point of an arbitrary dimension, algorithms have been provided by Rousseeuw and Ruts (1996) and Rousseeuw and Struyf (1998) for the halfspace depth, and by Dyckerhoff et al. (1996) for the zonoid depth. Aloupis (2006) gives a survey of algorithms for calculating different notions of medians and depths. But, calculating a depth trimmed region appears to be a much more demanding task. So far, algorithms have been constructed for the halfspace trimmed regions by Ruts and Rousseeuw (1996) and Miller et al. (2003) in

dimension 2, by Fukuda and Rosta (2004) and Liu et al. (2014) in arbitrary dimension. For bivariate zonoid trimmed regions, Dyckerhoff (2000) provides an algorithm that employs a circular sequence; Cascos (2007) uses the same approach for bivariate regions that are trimmed by the expected convex hull depth.

In this chapter we present an exact algorithm that efficiently calculates zonoid regions of any dimension. In contrast to most classical statistical tools, zonoid regions are genuine geometric notions. Consequently, our algorithm makes ample use of tools from computational geometry.

Consider an empirical distribution that gives probability $\frac{1}{n}$ to each of the observations $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and let $n \geq d$. The zonoid regions of the empirical distribution are defined by

$$D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \left\{ \frac{1}{n\alpha} \sum_{i=1}^{n} \lambda_i \mathbf{x}_i : \sum_{i=1}^{n} \lambda_i = n\alpha, 0 \leq \lambda \leq 1 \ \ \forall i \right\}, \tag{2.1}$$

$0 < \alpha < 1$. Immediately it can be seen from the definition that $D_{\frac{1}{n}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the convex hull of the data $\mathbf{x}_1, \ldots, \mathbf{x}_n$, while $D_1(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the set that contains the mean $\frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ as a single point. At $0 < \alpha < 1$, $D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is a convex polytope that lies between the convex hull and the expectation point and decreases strictly with α. The border of such polytope consists of a finite number of facets. Each facet is part of a hyperplane in $\mathbb{R}^d$ and can be described by the direction of its normal vector and its distance from the origin, that is to say, by some element $\mathbf{p}$ of the unit sphere S^{d-1} and some $\lambda \in \mathbb{R}_+$. The main task is to identify, from all directions $\mathbf{p} \in S^{d-1}$, those directions that determine the facets and calculate them in an efficient way.

The algorithm constructs the zonoid regions of an empirical distribution by their facets. Thus, for any data $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and any $\alpha \in [0, 1[$, the facets and vertices of $D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ are calculated and their coordinates are given. Our algorithm is efficient in that it computes the facets one after the other, proceeding from one facet to its neighbors.

In the dimension $d = 2$, Dyckerhoff (2000) has developed an algorithm for constructing zonoid regions. His procedure is based on the idea of a circular sequence (cf. Edelsbrunner (1987)): A ray starting at the center is turned like a clock's hand and the data points are projected onto this ray. However, the method of circular sequence works only with bivariate data. There is no obvious generalization of such a sequence to higher dimensions.

Our first task is to characterize the vertices and facets of a given zonoid region, given data points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and α. For this, we introduce a global structure that partitions $\mathbb{R}^d$ into direction cones that correspond one-to-one to the vertices of the zonoid region. In

this cone structure, the adjacency of vertices is investigated and characterized. A linear program is constructed to decide whether two vertices are neighbors. The resulting adjacency graph consists of elementary cycles that have either three or six nodes. Then we show that each facet of the zonoid region corresponds to exactly d data points and characterize the facet by a linear restriction on d data points.

Our second task is to put the facets of the zonoid region into an order according to which they can be efficiently calculated. For a given facet, a "jump-to-neighbor" procedure is introduced to transfer the calculation to the neighboring facets. Finally, a facet transversal graph is constructed, and a spanning tree order is realized to transverse this graph in an efficient way. This completes the algorithm.

Overview of this chapter: Section 2.2 presents zonoid regions of general probability distributions and surveys their principal statistical properties. In Section 2.3 the set of supporting vectors that belong to a given vertex of the zonoid is investigated. In Section 2.4 a global structure of direction cones is set up, and the adjacency of vertices is described through conditions on these direction cones. Section 2.5 provides a linear program by which the adjacency of vertices can be checked. Section 2.6 presents the adjacency graph and a characterization of facets of the zonoid region. In Section 2.7 the "jump-to-neighbor" procedure and the spanning tree order are introduced, according to which all facets are transversed. Section 2.8 concludes with a discussion of the complexity of our algorithm and its use in calculating zonoid regions for different α. This section also provides numerical experience and remarks on possible modifications of the algorithm. In the last Section 2.9 a formal algorithm is given.

2.2 Zonoid regions

Given a d-variate probability distribution function F, a family $\{D_\alpha(F)\}$ of sets in d-space, called *zonoid regions*, is defined as follows: $D_0(F) = \mathbb{R}^d$ and for $\alpha \in]0, 1]$

$$D_\alpha(F) \;=\; \left\{ \int_{\mathbb{R}^d} \mathbf{x}\, g(\mathbf{x})\, dF(\mathbf{x}) : 0 \le g \le \frac{1}{\alpha}\,, \quad \int_{\mathbb{R}^d} g(\mathbf{x})\, dF(\mathbf{x}) = 1 \right\}. \qquad (2.2)$$

For $\alpha \in]0, 1]$, these regions exist if and only if F has a finite expectation vector $\mu_F = \int_{\mathbb{R}^d} \mathbf{x} dF(\mathbf{x})$. It is obvious from the definition (2.2) that the zonoid regions are nested; the smallest region being the singleton set $D_1(F) = \{\mu_F\}$. Furthermore, each $D_\alpha(F)$ is bounded, closed, and convex. For an empirical distribution F, with equal mass on (not necessarily different) points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the definition specializes to the above definition (2.1).

In this section, we list a few principal properties of zonoid regions, which make them useful for statistical description and inference. For details and many other theoretical results, the reader is referred to Mosler (2002), as well as for applications to the multivariate analysis of location, dispersion and dependency.

Firstly, for every $\alpha \in [0,1]$, D_α is affine equivariant, i.e.,

$$D_\alpha(F_{\mathbf{XA}+\mathbf{c}}) = D_\alpha(F_X)\mathbf{A} + \mathbf{c} \quad \text{for any } d \times d \text{ matrix } \mathbf{A} \text{ having full rank,} \tag{2.3}$$

$\mathbf{c} \in \mathbb{R}^d$, and F having finite first moment. Hence, any statistical procedure based on zonoid regions is an *affine equivariant procedure*.

Secondly, the zonoid regions contain *full information* about the underlying distribution: For any two d-variate distribution functions F and G that have finite first moments, it holds

$$F = G \qquad \text{if and only if} \qquad D_\alpha(F) = D_\alpha(G) \quad \text{for all } \alpha \in]0,1]\,. \tag{2.4}$$

The uniqueness property (2.4) implies that any claim about a distribution F can be equivalently formulated as a claim about the zonoid regions of F and, thus, can be analyzed by geometric means.

Thirdly, zonoid regions lend themselves easily to *projection methods*: By projecting the zonoid regions of a distribution F onto some lower-dimensional subspace of $\mathbb{R}^d$, the zonoid regions of the projected distribution are obtained. In particular, any marginal of F has zonoid regions that are obtained by projection onto the respective coordinate space. See also Dyckerhoff (2004).

The fourth property is *continuity*: The zonoid region $D_\alpha(F)$ is continuous on α as well as on F. More precisely (see Th. 3.10 in Mosler (2002)), given a distribution F and a sequence (α_n) in $]0,1]$ that converges to $\alpha > 0$, it holds

$$D_{\alpha_n}(F) \quad \xrightarrow{H} \quad D_\alpha(F)\,, \tag{2.5}$$

Also, given $\alpha \in]0,1]$ and a sequence $F^{(n)}$ of distributions that is uniformly integrable and weakly convergent to F, it holds

$$D_\alpha(F^{(n)}) \quad \xrightarrow{H} \quad D_\alpha(F)\,. \tag{2.6}$$

In (2.5) and (2.6), $\xrightarrow{H}$ means convergence with respect to Hausdorff distance. (The Hausdorff distance of two compacts C and D is the smallest ϵ for which C plus an ϵ-ball includes D and D plus an ϵ-ball includes C as well.)

Fifthly, zonoid regions satisfy a *Law of Large Numbers*, which serves as the basis of statistical inference: Let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ denote an i.i.d. sample, with $\mathbf{X}_i \sim F$. For every α, it holds

$$D_\alpha(\mathbf{X}_1, \ldots \mathbf{X}_n) \xrightarrow{H} D_\alpha(F) \quad \text{almost surely}\,. \tag{2.7}$$

Thus, given an i.i.d. sample $\mathbf{X}_1, \ldots, \mathbf{X}_n$, the zonoid region $D_\alpha(\mathbf{X}_1, \ldots \mathbf{X}_n)$ serves as a *set-valued statistic* that estimates $D_\alpha(F)$. So far, the practical application of the zonoid region statistic was confined to the dimension $d = 2$, since no algorithm existed to calculate it for higher dimensions. In the rest of this chapter we develop an exact algorithm that works for any dimension $d \geq 3$.

2.3 Vertices and direction domains of a zonoid region

Let us first recall some standard notions and facts about convex sets and polytopes in $\mathbb{R}^d$. A *convex polytope* is the convex hull of a finite number of points or, equivalently, a bounded nonempty intersection of a finite number of closed halfspaces. A nonempty intersection of its boundary with a hyperplane is called a *facet* if it has an affine dimension $d-1$, and a *ridge* if it has an affine dimension $d-2$. It is called an *edge* if it is a line segment, and a *vertex* if it is a single point. The boundary of a convex polytope is the union of its facets. A convex polytope has a finite number of facets, ridges, edges, and vertices. An edge is the intersection of (at least) two facets, and a vertex is the intersection of (at least) two edges, $d-1$ ridges and d facets. A *hyperline* is an affine subspace of $\mathbb{R}^d$ that has a dimension 1.

A compact convex set is mentioned as a *convex body*. In particular, as a zonoid region is a bounded convex polytope, it forms a convex body in $\mathbb{R}^d$. The *support function* $h_C : S^{d-1} \to \mathbb{R}$ of a convex body $C \subset \mathbb{R}^d$ is defined by

$$h_C(\mathbf{p}) = \max\left\{\mathbf{p}'\mathbf{x} : \mathbf{x} \in C\right\}.$$

The support function of a convex body is closely related to its extreme points: A point $\mathbf{x}_0$ is extreme in C if and only if some $\mathbf{p} \in S^{d-1}$ exists so that

$$\mathbf{p}'\mathbf{x} = h_C(\mathbf{p}) \quad \text{implies} \quad \mathbf{x} = \mathbf{x}_0\,.$$

Now, for the given data $\mathbf{x}_1, \ldots, \mathbf{x}_n$, denote

$$H = H(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \{\mathbf{p} \in S^{d-1} : \mathbf{p}'\mathbf{x}_i = \mathbf{p}'\mathbf{x}_j \quad \text{for some } \ i \neq j\}\,.$$

Given a direction $\mathbf{p} \in S^{d-1} \backslash H$, the inner product $\mathbf{p}'\mathbf{x}$ projects the data points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ onto numbers $\mathbf{p}'\mathbf{x}_1, \ldots, \mathbf{p}'\mathbf{x}_n \in \mathbb{R}$. While the data as points in $\mathbb{R}^d$ have no natural total order, their projection does have. Thus, each $\mathbf{p} \in S^{d-1} \setminus H$ induces a total ordering of the data, i.e., a permutation $\pi_{\mathbf{p}}$ of the index set $1, \ldots, n$ given by

$$\mathbf{p}'\mathbf{x}_{\pi_{\mathbf{p}}(1)} < \mathbf{p}'\mathbf{x}_{\pi_{\mathbf{p}}(2)} < \cdots < \mathbf{p}'\mathbf{x}_{\pi_{\mathbf{p}}(n)} \,. \tag{2.8}$$

In the sequel we notate the **p**-ordered data by

$$\mathbf{X}^{\mathbf{p}} = (\mathbf{x}_1^{\mathbf{p}}, \ldots, \mathbf{x}_n^{\mathbf{p}}) \quad \text{with} \;\; \mathbf{x}_i^{\mathbf{p}} = \mathbf{x}_{\pi_{\mathbf{p}}(i)} \,, \quad i = 1, \ldots, n \,.$$

Proposition 2.1. (Dyckerhoff, 2000) *Let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ be pairwise distinct, $d \in \mathbb{N}$. For any $\mathbf{p} \in S^{d-1} \setminus H$ define*

$$\mathbf{x}_{\mathbf{p},\alpha} = \frac{1}{n\alpha} \sum_{i=1}^{n} \lambda_i^{\mathbf{p}} \mathbf{x}_i \,, \tag{2.9}$$

where

$$\lambda_i^{\mathbf{p}} = \begin{cases} 1 & \text{if } \pi_{\mathbf{p}}(i) > n - [n\alpha] \,, \\ n\alpha - [n\alpha] & \text{if } \pi_{\mathbf{p}}(i) = n - [n\alpha] \,, \\ 0 & \text{if } \pi_{\mathbf{p}}(i) < n - [n\alpha] \,. \end{cases}$$

Then the set of vertices of the zonoid region D_α is given by

$$\mathcal{V}(D_\alpha) = \{\mathbf{x}_{\mathbf{p},\alpha} \in \mathbb{R}^d : \mathbf{p} \in S^{d-1} \setminus H\} \,.$$

The set of all directions that yield vertices (= extreme points) of D_α is $S^{d-1} \setminus H$. Let $S(\mathbf{v}) \subset S^{d-1} \setminus H$ denote the subset of those directions that belong to a given vertex $\mathbf{v} \in S^{d-1} \setminus H$. $S(\mathbf{v})$ is named a *direction domain*. According to Proposition 2.1, all directions that provide the same permutation of the data belong to the same direction domain, i.e. to some common vertex $\mathbf{v}$. The family of direction domains $S(\mathbf{v}), \mathbf{v} \in \mathcal{V}(D_\alpha)$, forms a finite partition of $S^{d-1} \setminus H$.

Thus, the proposition yields a discretization of the continuum of possible directions of the vector $\mathbf{p}$, where the cardinality of the set of direction domains equals the number of vertices of the zonoid region. Thus, a one-to-one relation between domains of directions and vertices has been established.

In the sequel we assume that the data are *in general position*, i.e., every subset of $k+1$ data points generates an affine space of the dimension k, $k = 1, \ldots, d-1$. (If the data are not in general position, the subsequent discussion and the algorithm need to be modified, e.g., by slightly perturbing the data.) Also, without loss of generality, we shall assume that the mean of the data is at the origin, $\frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i = 0$.

2.4 Adjacent vertices

In this section we investigate the transition of one vertex to another, that is to say, of one direction domain to another. In our procedure we let a support vector $\mathbf{p}$ – that represents direction – continuously rotate on the unit sphere S^{d-1}. We start with an arbitrary $\mathbf{p} \in S^{d-1}$, which provides an initial permutation of the data points. As was mentioned, all $\mathbf{p}$ that produce the same permutation of points form a direction domain that belongs to a common vertex. When searching for all vertices, it obviously suffices to traverse each direction domain once. To do this, we shall characterize and identify the possible transitions of one domain to a neighboring one. Our identification procedure is based on the following observations:

The vector $\mathbf{p}$ *hits the boundary of a direction domain only if, for some* $i \neq j$, $\mathbf{p}'\mathbf{x}_i = \mathbf{p}'\mathbf{x}_j$ *holds, i.e.,* $\mathbf{p}$ *is orthogonal to* $\mathbf{x}_i - \mathbf{x}_j$.

Note that the pair $(\mathbf{p}'\mathbf{x}_i, \mathbf{p}'\mathbf{x}_j)$ is not unique. However, at most $d-1$ such pairs can arise, as the space of all vectors that are orthogonal to $\mathbf{p}$ has an affine dimension $d-1$, and the data are in general position.

The vector $\mathbf{p}$ *crosses the boundary of a direction domain only if, for some* $i \neq j$, $\mathbf{p}'\mathbf{x}_i$ *and* $\mathbf{p}'\mathbf{x}_j$ *change their order. That is to say, any transition from one permutation to another is done by swapping one of the pairs of data points.*

With Proposition 2.1 follows:

Theorem 2.2. (Identification of vertices) *The vector* $\mathbf{p}$ *passes from one direction domain (and one vertex) to a neighboring one if and only if* i *and* j *exist,* $i \neq j$, *so that* $\mathbf{p}$ *is orthogonal to* $\mathbf{x}_i - \mathbf{x}_j$, $\pi_{\mathbf{p}}(i) = n - [n \cdot \alpha]$, *and* $|\pi_{\mathbf{p}}(i) - \pi_{\mathbf{p}}(j)| = 1$.

Theorem 2.2 provides the basis for an algorithm that calculates all vertices of the zonoid region.

So far, we have considered parts of the unit sphere; the direction domains. They correspond to the vertices of the zonoid region. In the following discussion we will use the corresponding closed cones: For $\mathbf{v} \in \mathcal{V}(D_\alpha)$ define the *direction cone* $C(\mathbf{v})$,

$$C(\mathbf{v}) = \mathrm{cl}\{\lambda\mathbf{y} : \lambda \in \mathbb{R}_+, \mathbf{y} \in S(\mathbf{v})\}\,,$$

where $\mathrm{cl}(U)$ means closure of a set $U \in \mathbb{R}^d$. Then each $C(\mathbf{v})$ is a closed convex cone in $\mathbb{R}^d$ with an apex at the origin. It is finitely generated (i.e. it consists of all non-negative linear combinations of a finite number of vectors), and has a maximum of $n-1$ facets. The normals of its facets are described in Theorem 2.2.

The family of direction cones provides a global structure that divides the space $\mathbb{R}^d$ into sets corresponding to the vertices of D_α.

Now, consider three data points $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ and the hyperplanes through the origin that are orthogonal to $\mathbf{x}_i - \mathbf{x}_j, \mathbf{x}_i - \mathbf{x}_k$, and $\mathbf{x}_j - \mathbf{x}_k$, respectively. These hyperplanes intersect at a common hyperline that possibly contains a ridge of a direction cone. On the other hand, every ridge of a direction cone is contained in such a hyperline for some i, j, and k.

We can conclude: *In the global structure a ridge of a direction cone belongs to another direction cone only if it is a ridge of the latter, too.*

In the sequel we shall say that two direction cones are *adjacent cones* if they have a full facet in common. Turning the vector $\mathbf{p}$ through a boundary facet implies a transition from a direction cone to an adjacent one. It means that, to leave a current cone and enter an adjacent one, we have to move the vector $\mathbf{p}$ in an arbitrary way beyond one of the hyperplanes that carry the facets of the current cone.

2.5 A linear program for constructing adjacent vertices

Our next task is to find an explicit way of constructing all vertices that are neighbors of a given vertex $\mathbf{v}$. In other words, we will explicitly determine all direction cones that are adjacent to a given direction cone $C(\mathbf{v})$. These neighbors correspond to the facets of $C(\mathbf{v})$. Every facet of the cone is defined by a hyperplane. The set of hyperplanes that determine the cone's boundary is a subset of the hyperplanes described in Section 2.3. Each of these $n-1$ hyperplanes is represented by its "inner" normal $\mathbf{z}_j$, i.e., the normal pointing inside the cone:

$$\mathbf{z}_j = \begin{cases} \mathbf{x}^{\mathbf{p}}_{n-[n\alpha]} - \mathbf{x}^{\mathbf{p}}_j & \text{if } \; j = 1, \ldots, n - [n\alpha] - 1\,, \\ \mathbf{x}^{\mathbf{p}}_j - \mathbf{x}^{\mathbf{p}}_{n-[n\alpha]} & \text{if } \; j = n - [n\alpha] + 1, \ldots, n\,. \end{cases} \tag{2.10}$$

The $n-1$ normals are codirected with all directions $\mathbf{p}$ in the cone, i.e., they have non-negative inner products. In fact, $C(\mathbf{v})$ is the intersection of all corresponding halfspaces. In other words, $C(\mathbf{v})$ is the set of all vectors $\mathbf{p}$ that are codirected with the normals (2.10),

$$C(\mathbf{v}) = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{z}_j'\mathbf{p} \geq 0 \quad \text{for all } \; j = 1, \ldots, n-1\}\,.$$

Our task is to identify those hyperplanes (or, equivalently, their normals) that belong to the boundary of the cone $C(\mathbf{v})$. To determine whether $\mathbf{z}_j$ is a boundary normal, we

shall solve the following minimization problem:

$$\begin{aligned} & \mathbf{z}_j'\mathbf{p} \to \min\,, \\ \text{s.t.} \quad & \mathbf{p} \in C(\mathbf{v})\,, \\ & \sum_{i=1}^{d} |\mathbf{p}_i| = 1\,. \end{aligned} \tag{2.11}$$

If (2.11) has a positive minimal value, no support vector, $\mathbf{p} \in C(\mathbf{v})$, exists that is orthogonal to $\mathbf{z}_j$. Hence, $\mathbf{z}_j$ is not a boundary normal of $C(\mathbf{v})$. If (2.11) is minimized with value $\mathbf{z}_j'\mathbf{p}^* = 0$, we can conclude that $\mathbf{p}^*$ is a support vector that belongs to the boundary of $C(\mathbf{v})$ and is an element of the hyperplane that has normal $\mathbf{z}_j$. Consequently, $\mathbf{p}^*$ also belongs to the boundary of the direction cone $C(\tilde{\mathbf{v}})$ of some vertex $\tilde{\mathbf{v}}$ that borders on the current vertex $\mathbf{v}$.

(2.11) can be rewritten as a linear program (LP_j),

$$\begin{aligned} & \mathbf{z}_j'(\mathbf{p}^+ - \mathbf{p}^-) \to \min\,, \\ \text{s.t.} \quad & \mathbf{z}_j'(\mathbf{p}^+ - \mathbf{p}^-) \geq 0 \quad j = 1, \ldots, n-1\,, \\ & \sum_{i=1}^{d} (p_i^+ + p_i^-) = 1\,, \\ & p_i^+ \geq 0\,, \quad p_i^- \geq 0\,, \quad i = 1, \ldots, d\,. \end{aligned} \tag{2.12}$$

Here we have inserted $\mathbf{p} = \mathbf{p}^+ - \mathbf{p}^-$, where $\mathbf{p}^+ = (p_1^+, \ldots, p_d^+)'$ and $\mathbf{p}^- = (p_1^-, \ldots, p_d^-)'$ are the positive and negative parts of $\mathbf{p}$, respectively. The linear program (2.12) is solved by the simplex method.

To find all vertices bordering on $\mathbf{v}$, we may solve the linear programs (2.12) for $j = 1, \ldots, n-1$. As all programs have the same set of feasible solutions, the calculations can be shortened by solving them simultaneously. In fact, the number of neighbors is small compared to $n - 1$. Therefore, the number of basic feasible solutions in the simplex method will be relatively small, too, which leads to a high average efficiency of the simplex method. If n is large, the dual simplex method may outperform the primal approach.

Each ridge of a direction cone is an intersection of three hyperplanes. On the ridge either three or six direction cones touch each other. Let us consider these two cases in more detail:

Remember that, given a vertex $\mathbf{v}$, for all $\mathbf{p} \in C(\mathbf{v})$ the point $\mathbf{x}^{\mathbf{p}}_{n-[n\alpha]}$ does not depend on $\mathbf{p}$. We call this point the *main point* of $C(\mathbf{v})$.

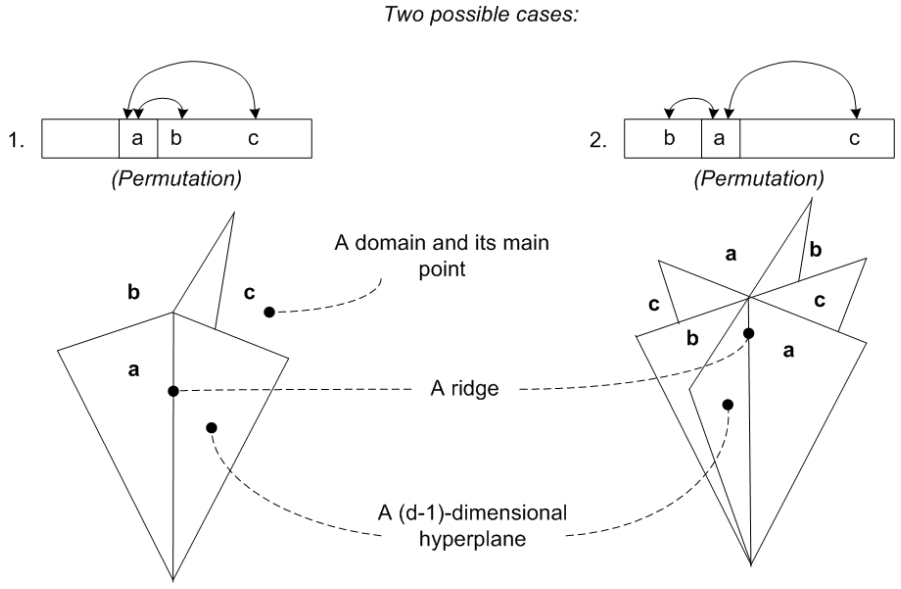

FIGURE 2.1: Neighboring cones near the common ridge.

Consider three direction cones $C(\mathbf{v}), C(\mathbf{w})$, and $C(\mathbf{u})$ that have a common ridge and denote their main points by $\mathbf{a}, \mathbf{b}, \mathbf{c}$, respectively. For all $\mathbf{p} \in C(\mathbf{v})$ these three main points are $\mathbf{p}$-ordered in the same way, either with $\mathbf{a}$ in the middle or not. If $\mathbf{a}$ is in the middle, $\mathbf{b}$ and $\mathbf{c}$ cannot switch their positions. Hence, $C(\mathbf{w})$ and $C(\mathbf{u})$ have no boundary hyperplane in common, and $\mathbf{w}$ and $\mathbf{u}$ are not adjacent vertices. In this case, a total of six direction cones meet at the common ridge. If, on the other hand, $\mathbf{b}$ and $\mathbf{c}$ are on the same $\mathbf{p}$-side of $\mathbf{a}$, their positions can switch and $\mathbf{w}$ and $\mathbf{u}$ are neighboring vertices. In this case, only three direction cones unite at the common ridge. The two cases are illustrated in Figure 2.1.

2.6 Edges and facets

Having obtained an efficient procedure for finding extreme points we now must create an efficient one for constructing all facets of the zonoid region. Next we characterize the edges of the zonoid region.

Lemma 2.3 (Vertices and edges). *Let $C(\mathbf{v})$ and $C(\mathbf{w})$ be direction cones. The line connecting $\mathbf{v}$ and $\mathbf{w}$ is an edge if and only if $C(\mathbf{v})$ and $C(\mathbf{w})$ are adjacent cones.*

Proof. Recall that the zonoid region D_α is a convex polytope, and its extreme points form the vertices of this polytope. As $C(\mathbf{v})$ and $C(\mathbf{w})$ are direction cones of vertices $\mathbf{v}$ and $\mathbf{w}$, for all $\mathbf{x} \in D_\alpha$ it holds that:

$$\mathbf{p}'\mathbf{x} \leq \mathbf{p}'\mathbf{v} \quad \text{if } \mathbf{p} \in C(\mathbf{v}), \quad \text{and} \quad \mathbf{p}'\mathbf{x} \leq \mathbf{p}'\mathbf{w} \quad \text{if } \mathbf{p} \in C(\mathbf{w}). \tag{2.13}$$

Now assume that $C(\mathbf{v})$ and $C(\mathbf{w})$ are adjacent cones. Then for each $\mathbf{p}$ in their common boundary $C(\mathbf{v}) \cap C(\mathbf{w})$ it holds that $\mathbf{p}'\mathbf{v} = \mathbf{p}'\mathbf{w}$. Hence, for all $\mathbf{x} \in D_\alpha$

$$\mathbf{p}'\mathbf{x} \leq \mathbf{p}'(\lambda\mathbf{v} + (1-\lambda)\mathbf{w}) \quad \text{if } \lambda \in [0,1]\,. \tag{2.14}$$

It follows that the line connecting $\mathbf{v}$ and $\mathbf{w}$ is an edge of the polytope. On the other hand, assume that this line $\overline{\mathbf{vw}}$ is an edge. Then some $\mathbf{p}$ exists that, for all $\mathbf{x} \in D_\alpha$, satisfies (2.14). For this $\mathbf{p}$, it must hold that $\mathbf{p}'\mathbf{v} = \mathbf{p}'\mathbf{w}$. In view of (2.13), we can conclude that (2.14) is true for all $\mathbf{x} \in D_\alpha$ if and only if

$$\mathbf{p}'(\mathbf{v}-\mathbf{w}) = 0\,,\ \mathbf{p}'\mathbf{v} \geq 0\,,\ \text{and}\ \ \mathbf{p}'\mathbf{w} \geq 0\,,$$

that means, $\mathbf{p}$ is in a $d-1$-dimensional cone which is a subset of $C(\mathbf{v})$ and of $C(\mathbf{w})$. Consequently, $C(\mathbf{v})$ and $C(\mathbf{w})$ have a full facet in common, and are, thus, adjacent cones. □

Lemma 2.3 provides a unique correspondence between the adjacency of direction cones in the global cone structure and the existence of edges of the zonoid region.

Recall that adjacent direction cones are cones that have a common facet. The adjacency information of the zonoid region, which is a polytope, is represented by its *adjacency graph*, which consists of the polytope's vertices and edges. Above, we have demonstrated that either three or six direction cones touch each other on a ridge. Hence the adjacency graph is a concatenation of elementary cycles, each connecting either three or six vertices. This is illustrated in Figure 2.2:

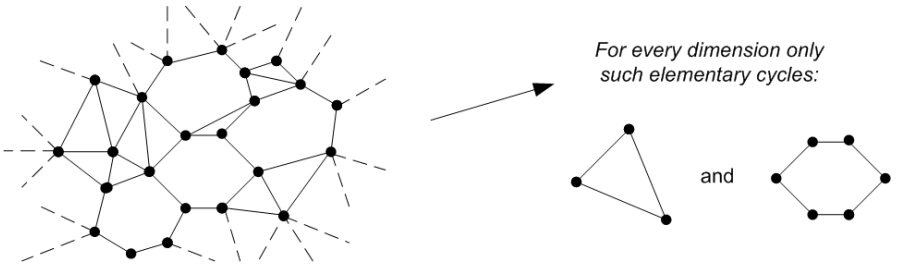

FIGURE 2.2: The structure of the adjacency graph.

Now, consider a facet of the zonoid region. Let the vector $\mathbf{p}$ be directed orthogonally to the facet. Then, if points from the data cloud are in general position, exactly d points $\mathbf{x}_{\pi_\mathbf{p}(k)}, \mathbf{x}_{\pi_\mathbf{p}(k+1)}, \ldots, \mathbf{x}_{\pi_\mathbf{p}(k+d-1)}$ exist so that:

$$\left.\begin{array}{l} \mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k)} = \mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k+1)} = \cdots = \mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k+d-1)} \\ \text{with some } k\,, \quad k \leq n - [n\alpha] \leq k+d-1\,. \end{array}\right\} \tag{2.15}$$

Obviously, the indices $k, \ldots, k+d-1$ in (2.15) are not unique. However, any permutation of these d points yields the same facet. We may conclude the following theorem.

Theorem 2.4. (Identification of facets) *Each facet can be identified by a set of exactly d points from the data cloud and one of its vertices. Moreover, if there is a support vector that defines a permutation satisfying (2.15), then this permutation and these d data points define a facet of the zonoid region.*

It can easily be seen from Theorem 2.4 that, if $k < n - [n\alpha] < k + d - 1$, the facet can have more than d vertices . A facet will be mentioned as *redundant* if it has more than d vertices, and as *non-redundant* if it has exactly d vertices. In any case, there are only d different main points that belong to a facet. Let $\ell = n - [n\alpha] - k$. We obtain:

Corollary 2.5. *The number of vertices of the facet equals $d \cdot \binom{d-1}{\ell}$.*

Proof. As stated above, the total number of possible relative positions of d points is $d!$. But, according to (2.9), the relative position of points in

$$[k, (n - [n\alpha])[\quad \text{and} \quad](n - [n\alpha]), k + d - 1]$$

is not significant, i.e. $\binom{d-1}{\ell}$ different cases remain. This number is multiplied by d, which is the number of possible main points. □

Corollary 2.6. *Each set of d points from the data cloud defines, at most, one facet of a zonoid region unless:*

$$k \leq n - [n\alpha] \leq k + d - 1 \quad \textit{and} \quad k \leq n - [n - n\alpha] \leq k + d - 1\,. \tag{2.16}$$

Otherwise it defines exactly two parallel facets.

Proof. The first statement is clear from Theorem 2.4. The second is based on the fact that, if the condition (2.16) is met, then the inverted support vector also defines a permutation satisfying (2.15). □

Corollary 2.7. *For each set of d data points some α exists, so that the set defines a facet of the zonoid α-region.*

Proof. In fact, for a set of exactly d data points it is always possible to find α so that the condition (2.15) is met. Then the statement follows from Theorem 2.4. □

In the case of a *non-redundant facet* we have $\ell = 0$. Then, according to Corollary 2.5 the facet is a $(d-1)$-dimensional simplex having d vertices. The vertices identify the facet;

they are pairwise adjacent and correspond to the pairwise adjacent direction cones. In turn, every set of d vertices that correspond to pairwise neighboring cones defines either a facet or a cut of the zonoid region. Thus, also in this case, the identification of the facet is based on the adjacency graph. Based on Corollary 2.6 we can generate an arbitrary facet as follows:

1. Choose an arbitrary set of d points.
2. Check whether this set defines a facet. If not, go back.
3. Create the corresponding facet.

We will also use this procedure to initialize our algorithm by creating a first facet. Thus, Theorem 2.4 and its corollaries provide a procedure for identifying each facet of the zonoid region.

2.7 Sequencing the facets

To complete the algorithm, we have to create a procedure that generates all facets in a sequential way. For this, we specify a total ordering of the set of facets. In the case $d = 2$ such an order is easily created by a circular sequence; see Dyckerhoff (2000). In the dimension $d \geq 3$ we can solve this problem by introducing a *spanning tree order (STO)*. Consider a given facet. Each ridge of it corresponds to exactly one neighboring facet and all neighboring facets can be found by passing through the ridges. The following theorem and its proof tell us the number of adjacent faces, that is the number of ridges, and how the ridges are obtained.

Theorem 2.8. (Neighboring facets)

(i) A facet has either $2d$ or d neighboring facets.

(ii) It has d neighbors if and only if it is non-redundant.

Proof. Let the support vector $\mathbf{p}$ be orthogonal to the facet and condition (2.15) be met. A minimal violation of orthogonality is achieved by an infinitesimal move of $\mathbf{p}$ in a direction that is perpendicular to a ridge of the facet and non-perpendicular to its other ridges. This corresponds to the following change in the first equation of (2.15): Either $\mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k)} < \mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k+1)} = \cdots = \mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k+d-1)}$ or $\mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k)} = \cdots = \mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k+d-2)} < \mathbf{p}'\mathbf{x}_{\pi_\mathbf{p}(k+d-1)}$. That is, a ridge is obtained by removing one of the points that define the facet according to Theorem 2.4.

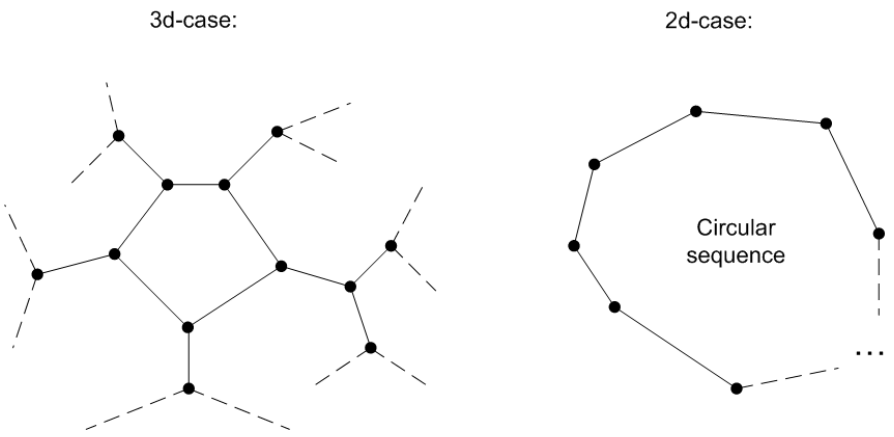

FIGURE 2.3: Examples of the facet traversal graph.

By removing one point to the higher and one to the lower part of the permutation, we obtain two ridges. Note that in the non-redundant case we can generate only one ridge in order not to violate the second equation of (2.15). The number of ridges is $2d$ (d in a non-redundant case), as there are d points which have to be removed. Obviously, the number of facets that neighbor the current one equals the number of its ridges. □

Thus, by removing one of the points that characterize the current facet according to the Theorem 2.4, we are directed to either one or two neighboring facets. If the current facet is redundant, removing a point yields two parallel ridges. Thus, we get two neighbors for a redundant, and one for a non-redundant facet.

Next, we search for a new adjacent facet that shares a given ridge. For this, rotate the support vector $\mathbf{p}$ in the plane orthogonal to the ridge defined by the $d-1$ points. Obviously, if $\mathbf{x}_i$ has been removed from the higher (resp. lower) part of the permutation, $\mathbf{p}$ has to be rotated in the direction of increasing (resp. decreasing) $\mathbf{p}'\mathbf{x}_i$. The rotation stops when the (2.15) first equation is met, that means, $\mathbf{p}$ has reached a normal of a new facet.

As this procedure produces a "jump" from the current facet to one of its neighbors, we shall mention it as the *"jump-to-neighbor" procedure*. According to this procedure, sequentially generated facets are identified in a similar way, which allows for an efficient implementation. Moreover, as the traversal through all neighbors guarantees the absence of "gaps", no facet will be lost.

Based on Theorem 2.8 we shall construct a special graph; the *facet traversal graph (FTG)*. The vertices of the FTG correspond to the set of all facets of the zonoid region, and the edges of the FTG indicate the neighborhood of facets. Each vertex of this graph joins either d or $2d$ edges.

A sequential procedure for determining the zonoid region consists in transversing all vertices of the corresponding FTG. Note, that in the dimension $d = 2$, the FTG is an elementary cycle and its traversal is trivial and unique. In fact $d = 2$ is a degenerated case. For $d \geq 3$ we construct a spanning tree order (STO) of the FTG, which orders the set of facets. By "jumps-to-neighbor" the STO is created in a dynamic way:

1. Organize a queue.
2. In each step, pop from the queue a current facet, which corresponds to a vertex of the FTG. Add to the queue all adjacent vertices of the current vertex that have not been processed so far. Mark the current vertex as processed.
3. Marking of the vertices is done through a hash table, where hash codes of all the processed vertices (i.e. facets of the zonoid region) are stored.

According to Corollary 2.6 the record for each facet in the hash table is fully described by d integer numbers. These numbers are the labels of main elements of d points that define the facet. If these points define two parallel facets, these facets can easily be generated in one step, thus making it possible to have one record for them in the hash table.

The linear order of the vertices (facets) is provided by its final positioning in the queue. The realization of the STO is illustrated by Figure 2.4.

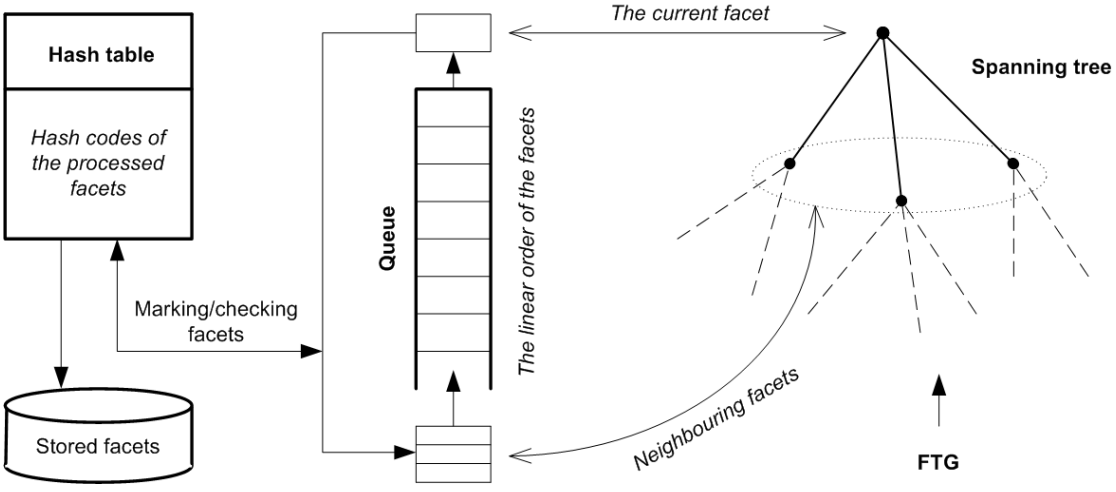

FIGURE 2.4: Realization of the STO.

So far, we have constructed an algorithm to compute $D\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ for a single given $\alpha \in]0, 1[$. Finally, this procedure is modified to efficiently calculate $D^*_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ for all α^* in an interval around α.

Given $\mathbf{p}_0$, consider the intervall A_k of those α whose main element has the same index k, that is $n - [n\alpha] = \pi_{\mathbf{p}_0}(k)$ or, equivalently,

$$\frac{n - \pi_{\mathbf{p}_0}(k)}{n} \leq \alpha < \frac{n - \pi_{\mathbf{p}_0}(k) + 1}{n} \,. \tag{2.17}$$

For all $\alpha \in A_k$, the global cone structure is the same. When we calculate the zonoid region for some $\alpha \in A_k$, we can simultaneously determine the zonoid regions for all $\alpha \in A_k$ by using the same global cone structure and only recalculating the distances of facets from the origin.

Note that that there are only n possible different global cone structures. Hence the global cone structure is fully determined by the initial position $\mathbf{p}_0$ of the support vector and the corresponding main point, which can be any of the n data points. Consequently, for determining the *whole family of zonoid regions*, $D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ for all α, it suffices to run the modified algorithm n times only.

2.8 Discussion

An exact algorithm has been constructed to compute the zonoid regions of an empirical distribution in d-space. It calculates all of the vertices, edges, and facets of a zonoid region at any given depth $\alpha \in]0, 1[$. (Recall that $\alpha = 0$ and $\alpha = 1$ are trivial cases.) This approach requires that the dimension $d - 2$ of ridges is not lower than 1, which is the dimension of edges. It works for any dimension $d \geq 3$ and any number n of data points.

A hash table plays a significant role, as it stores the vertices, once generated, in a special structure and facilitates a fast check of whether the vertex has already been processed. Each facet is generated only once. Thus the algorithm has as many loops as the zonoid region has facets. Obviously, this is the minimum number of facet generating loops in this sort of algorithm.

In a single facet generating loop, the most costly operations are as follows: Calculate the hyperplane equation of the current facet, calculate its distance from the origin and obtain the neighboring facets. This is done by solving linear equations and finding inner products only. The complexity of the first operation is $\mathcal{O}(d^3)$. Up to d such operations are performed in each loop. The complexity of getting $n - 1$ inner products is assessed $\mathcal{O}(nd)$. Hence, the complexity of one facet generating loop is described by $\mathcal{O}(d^2(d^2 + n))$.

The number of computational loops of the algorithm is equal to the number of facets of the zonoid region. If the average number of facets is denoted $N(n, d)$, the average computational complexity of the algorithm amounts to $\mathcal{O}(d^2(d^2 + n) \cdot N(n, d))$.

Note that the complexity increases only moderately with d. For example, consider two data clouds of dimensions d_1 and d_2 ($d_1 < d_2$) that contain the same number of points n. Then the second data cloud will form a polytope that has a more trivial structure in $\mathbb{R}^{d_2}$ than the first has in $\mathbb{R}^{d_1}$. It is also easy to see that there is no operation in the algorithm whose complexity grows exponentially with the growth of the dimension. Altogether the complexity is polynomial in n and d. This confirms the efficiency of our algorithm.

General memory resources are used, in the first place, for storing a hash table and created facets. Each facet occupies $\mathcal{O}(d)$ storage size, while a hash table in almost any case has a constant size C, not depending on n and d. Therefore, the use of general memory is of the order $\mathcal{O}(N(n, d) \cdot d + C)$. Facets, once they have been created, are put into a secondary store, thus considerably reducing the storage cost.

Figure 2.5 illustrates its application by exhibiting zonoid regions for a small data set of five points of the dimension three and for several values of α. Each zonoid region is depicted in three directions (by revolving it on a vertical axis). The data points are shown as little pyramids. (Note that these 3d-pictures employ a perspective view.)

The algorithm can be downloaded as an *R* package. It has been implemented on a standard PC. Table 2.1 exhibits, for different choices of d and n, average values of total time (in seconds), number of facets, and time per facet. Note that for all calculations the same $\alpha = 0.317$ was taken. As the number of facets depends on the data, the efficiency of the algorithm may be judged by its computation time per facet. Table 2.1 gives an idea of how the time for computing one facet grows with d and n. It appears that the "time per facet" increases polynomially with d; moreover, the increase is close to being linear. Concerning n, a tendency towards saturation at some constant value is indicated. The given results also suggest that the growth in the "number of facets" is polynomial as well. Hence we can suspect that the aggregate complexity is polynomial. These considerations are also confirmed by the genelized algorithm (cf. Section 3.4.2).

Much computational load can be spared when we simultaneously calculate zonoid regions for several α that are sufficiently close to each other. If $[n \cdot \alpha_1] = \ldots = [n \cdot \alpha_k]$ holds, complete facets have to be computed for α_1 only, while for $\alpha_2, \ldots, \alpha_k$ all facets are parallel to them; so, only their distances from the origin have to be calculated.

The algorithm as it stands generates the facets one after the other in a deterministic way. It may be modified in order to gradually cover certain specified parts of the zonoid region

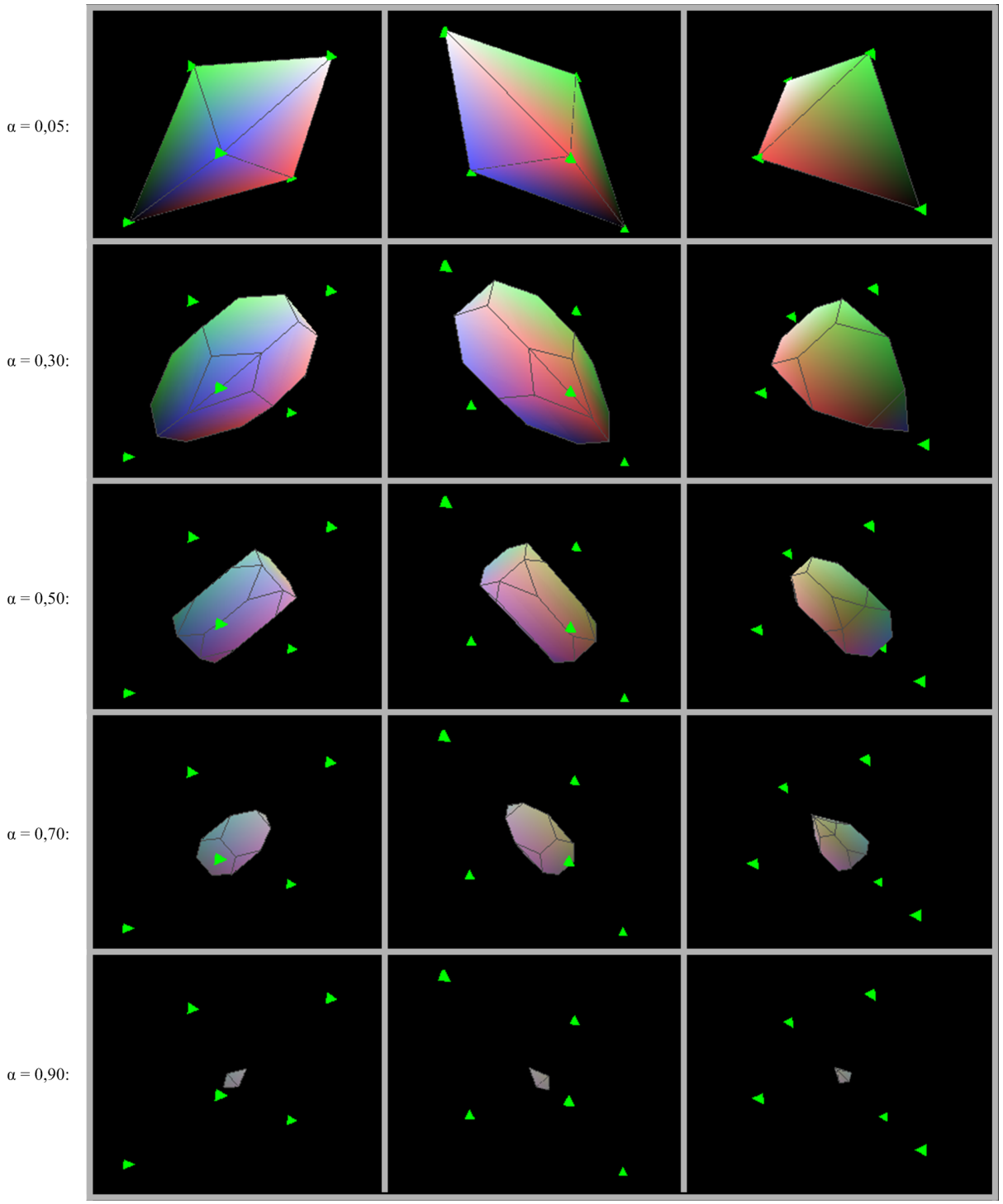

FIGURE 2.5: Zonoid regions of five points of the dimension three.

d	n	time per facet	number of facets	total time [sec]
3	10	0.002437	106	0.259
3	15	0.003022	241	0.722
3	20	0.003474	476	1.653
4	10	0.003142	257	0.809
4	15	0.004444	948	4.216
4	20	0.005163	2840	14.667
5	10	0.003714	377	1.403
5	15	0.006250	2556	15.980
5	20	0.007257	13082	95.133
6	10	0.004083	377	1.542
6	15	0.008112	5005	40.600
6	20	0.009385	38177	356.648

TABLE 2.1: First computational results of the zonoid regions algorithm.

that are of special interest. For instance, all facets belonging to the lower boundary may be constructed without generating the remaining facets of the zonoid region.

Moreover, the structure of the algorithm lends itself well to being parallelized on a high-performance cluster system.

To speed up the procedure, our exact algorithm can also be modified by imposing heuristic rules on the choice of adjacent facets. For instance, as a heuristic rule we may prefer redundant facets to non-redundant ones, since a redundant facet borders on more other facets and, thus, can be regarded as a facet that determines the zonoid region "more strictly" than others do. In any case, the exact algorithm serves as a benchmark procedure to be compared with any heuristic procedure, regarding precision as well as speed.

2.9 The algorithm

Input

d (dimension of the data space, $d \geq 3$)

n (number of data points, $n > d$)

cloud (data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$)

α (depth parameter)

Output

zonoid_region (all facets of zonoid region, with coordinates of their vertices)

Steps of the Algorithm

A. Initialization:

   a. Read the input.

   b. Calculate $\overline{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$.

   c. Substitute $\mathbf{x}_i - \overline{\mathbf{x}}$ for $\mathbf{x}_i$, $i = 1, \ldots, n$.

B. Determining a first facet:

   a. Choose from *cloud* a subset *d_set* that contains d data points.

   b. Calculate, to the hyperplane through these d points, a normal vector $\mathbf{r}$.

   c. Substitute $\mathbf{r}$ for $\mathbf{p}$ and consider the permutation $\pi_{\mathbf{p}}$.

   d. If *d_set* $= \{x_{\pi(k)}, \ldots, x_{\pi(k+d-1)}\}$ satisfies (2.15), *d_set* defines the first facet *ffacet*. Otherwise, go to B.a.

   e. Calculate a vertex of *ffacet* from (2.9) and the constant of the hyperplane equation.

   f. Place *ffacet* $\hookrightarrow$ *queue*.

   g. If condition (2.16) is met, generate *dualffacet*, which is parallel to *ffacet* and based on $\mathbf{p}_{inv} = -\mathbf{p}$. Place *dualffacet* $\hookrightarrow$ *queue*.

   h. Calculate the hash code of *ffacet* and place it $\hookrightarrow$ *hash_table*.

C. Determining all facets:

   a. Take *curr_facet* $\hookleftarrow$ front of the *queue*.

   b. Create neighboring facets based on each of the points *curr_point* that define *curr_facet*.

      I. Create ridges by removing *curr_point*.

         i. If $k < n - [n\alpha]$ in (2.15), obtain *lower_ridge* by removing *curr_point* from the lower part of $\pi_{\mathbf{p}}$.

         ii. If $k + d - 1 > n - [n\alpha]$ in (2.15), obtain *higher_ridge* by removing *curr_point* from the higher part of $\pi_{\mathbf{p}}$.

      II. Choose a vector $\mathbf{z}$ that is orthogonal to the found ridge and linearly independent of $\mathbf{p}$.

         A. Given a normal vector $\mathbf{r}$ to *curr_facet*, $\mathbf{z}$ is calculated by exchanging one equation in the linear system used for calculating $\mathbf{r}$.

         B. $\mathbf{p}$ and $\mathbf{z}$ define the basis B_2 of a plane.

      III. If *lower_ridge* has been created:

A. Rotate $\mathbf{p}$ in the plane that is generated by basis B_2. In doing this, start from $\mathbf{r}$ in such direction that the index of the point being removed decreases.

B. Stop if the first equation of (2.15) is met. Then, *new_point* has been found.

C. Substitute *current_facet* by *lower_new_facet*, and exchange *new_point* with *curr_point* the set of points defining the facet.

III'. If *higher_ridge* has been created:

i. Rotate $\mathbf{p}$ in the plane generated by basis B_2. In doing this, start from $\mathbf{r}$ in such direction that the index of the point being removed increases.

ii. Stop if the first equation of (2.15) is met. Then, *new_point* has been found.

iii. Substitute *current_facet* by *higher_new_facet*, and exchange *new_point* with *curr_point*.

IV. Calculate the hash codes of *lower_new_facet* and *higher_new_facet*. Check in *hash_table* whether these facets are new. If not, go to C.a.

V. For each of the created facets, calculate the vertices and the facet's equation by (2.9). Place the created facets $\hookrightarrow$ *queue*.

V'. If condition (2.16) is met, generate a parallel facet based on $\mathbf{p}_{inv} = -\mathbf{p}$. Place the facet $\hookrightarrow$ *queue*, too.

VI. Place hash codes $\hookrightarrow$ *hash_table*.

c. Shift *curr_facet* by $\overline{\mathbf{x}}$ and transfer it from *queue* to *zonoid_region*.

d. If *queue* is not empty, go to C.a. Otherwise, stop: Then, *zonoid_region* contains all facets of the zonoid region.

Some details of the algorithm:

1. Given a hyperplane through d points, a normal vector $\mathbf{r}$ is obtained as follows: Let $\mathbf{A}$ be the matrix that contains the given d vectors minus the first vector as rows and solve the linear system $\mathbf{A} \cdot \mathbf{r} = \mathbf{b}$.

2. The hash code is calculated by creating a bit row from d sorted integer numbers that correspond to the labels of main elements of the d points defining the facet.

3. The neighbors of the current vertex are found as follows:

   (a) Create a bundle of vectors that connect the current main point and all other points from *cloud*.

(b) Reverse all vectors in the bundle that correspond to points having lower rank than $n - [n\alpha]$ in the permutation.

(c) For each vector in the bundle,

i. Specify a linear program LP_j (2.12) for the current vector,

ii. Solve the specified linear program by the dual simplex method.

iii. If the objective is minimized with value 0, go to the next step. Otherwise, go to 3(c)i.

iv. The point from *cloud* that corresponds to this vector is the main point of adjacent direction cone.

v. Exchange the new main point with the current main point in the current permutation. This yields a new neighboring vertex.

Chapter 3

Weighted-Mean Trimmed Regions and Distortion Risks

Trimmed regions are a powerful tool of multivariate data analysis. They describe a probability distribution in Euclidean d-space regarding location, dispersion, and shape, and they order multivariate data with respect to their centrality. Dyckerhoff and Mosler (2011) have introduced the class of weighted-mean trimmed regions, which possess attractive properties regarding continuity, subadditivity, and monotonicity.

In this chapter we present an exact algorithm to compute the weighted-mean trimmed regions of a given data cloud in arbitrary dimension d. These trimmed regions are convex polytopes in $\mathbb{R}^d$. To calculate them, the algorithm builds on methods from computational geometry. We employ a special characterization of a region's facets, and extract information about the adjacency of the facets from the data. A key problem consists in ordering the facets. It is solved by the introduction of a tree-based order, by which the whole surface can be traversed efficiently with the minimal number of computations. We decribe the algorithm and its implementation in *C++* and *R* package WMTregions.

3.1 Motivation

Given d-variate data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, an *$\alpha$-trimmed region* $D_\alpha(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ is a convex compact set in $\mathbb{R}^d$ that depends on the data in an affine equivariant way, i.e., for every matrix $\mathbf{A} \in \mathbb{R}^{m \times d}$ and every $b \in \mathbb{R}^m$ it holds

$$D_\alpha(\mathbf{A}\mathbf{x}_1 + b, \ldots, \mathbf{A}\mathbf{x}_n + b) = \mathbf{A} D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) + b\,.$$

The parameter α varies in an interval such that the family $(D_\alpha(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n))_\alpha$ is nested decreasing in α, i.e., $\alpha < \beta$ implies $D_\beta(\mathbf{x}_1, \ldots, \mathbf{x}_n) \subset D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$. The smallest region is regarded as a particular median of the data.

Several special notions of trimmed regions have been introduced in the literature, among them the *Mahalanobis regions*, the *halfspace regions*, and the *zonoid regions*; for recent surveys, see Serfling (2006), Cascos (2009). Applications include multivariate data analysis (Liu et al., 1999), classification (Mosler and Hoberg, 2006), tests for multivariate location and scale (Dyckerhoff, 2002), risk measurement (Cascos and Molchanov, 2007), and many others. The various notions of trimmed regions differ in properties like continuity, robustness, and sensitivity regarding the data. Depending on the type of application different properties are relevant. E.g., Mahalanobis regions are ellipses around the mean of the data and based on their covariance matrix; by this they can neither reflect a possible asymmetry of the distribution nor characterize it in a unique way. Both halfspace regions and zonoid regions reflect asymmetries and characterize the distribution. Halfspace regions are more robust against outliers than zonoid regions; if robustness is an issue, the latter need some preprocessing of the data.

Dyckerhoff and Mosler (2011) have introduced the class of weighted-mean trimmed regions, which possess additional attractive properties and include the zonoid regions as a special case. Weighted-mean trimmed regions are continuous in the data as well as in the parameter, which means that both the function $(\mathbf{x}_1, \ldots, \mathbf{x}_n) \mapsto D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and the function $\alpha \mapsto D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ are continuous in terms of Hausdorff convergence of sets. Moreover, weighted-mean trimmed regions are subadditive and monotone, which properties have a substantial interpretation in terms of d-variate risk and allow the construction of set-valued risk measures that are coherent (Cascos and Molchanov, 2007).

To be useful in data applications, a notion of trimmed regions must be computable. Bivariate trimmed regions of any type can be calculated by constructing a *circular sequence*, like in Dyckerhoff (2000) and Cascos (2007), but only a few procedures are known in dimension $d > 2$. Mahalanobis regions are easily determined in any dimension d, as they only employ the mean and the dispersion matrix of the data. Mosler et al. (2009) develop an efficient geometric algorithm for zonoid regions in any dimension, and Hallin et al. (2010) provide a parametric linear program for calculating halfspace regions.

In this chapter we present an exact algorithm to compute the weighted-mean trimmed regions of a given data cloud in arbitrary dimension d. These trimmed regions are convex polytopes in $\mathbb{R}^d$. To calculate them, the algorithm builds on methods from combinatorial and computational geometry. A region's facet is characterized by $d-1$ pairs of data points. Based on them the normal (support vector) of the facet is determined and by

properly rotating the support vector an adjacent facet is found. A key problem consists in ordering the facets. It is solved by the introduction of a tree-based order.

Overview of the chapter: Section 3.2 provides a brief introduction into the notion of weighted-mean (WM) trimmed regions. The main results of the chapter are contained in Section 3.3, which presents the basic geometrical ideas of the algorithm, in particular the construction of a facet on the basis of $d-1$ data point differences and the transition to a neighboring facet by rotating the support vector and exchanging the basis. Section 3.4 provides a formal description of the algorithm with the analysis of its complexity. Section 3.5 delineates the *R* package and the program realization in *C++*, while Section 3.7 provides conclusions and a discussion of perspectives. The last Section 3.8 proposes heuristics for speeding up the algorithm.

3.2 Weighted-mean trimming

This section reviews the general notion of weighted-mean trimmed regions and two of its special cases, the zonoid regions and a modified version of the expected convex hull (ECH) regions - the ECH* regions.

3.2.1 Definition and principal properties

Weighted-mean trimmed regions are convex bodies in $\mathbb{R}^d$. Recall that a convex body $K \subset \mathbb{R}^d$ is uniquely determined by its support function h_K (see, e.g., Rockafellar (1997)),

$$h_K(\mathbf{p}) = \max\left\{\mathbf{p}'\mathbf{x} \,|\, \mathbf{x} \in K\right\}, \quad p \in S^{d-1},$$

where S^{d-1} denotes the unit sphere in $\mathbb{R}^d$.

To define the weighted-mean α-trimmed region of a given data cloud $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, we construct its support function as follows: For $\mathbf{p} \in S^{d-1}$, consider the subspace $\{\lambda p | \lambda \in \mathbb{R}\}$. By projecting the data on this subspace a linear ordering is obtained,

$$\mathbf{p}'\mathbf{x}_{\pi_p(1)} \leq \mathbf{p}'\mathbf{x}_{\pi_p(2)} \leq \cdots \leq \mathbf{p}'\mathbf{x}_{\pi_p(n)}\,, \tag{3.1}$$

and, by this, a permutation π_p of the indices $1, 2, \ldots, n$. Note that, if no equalities arise in (3.1), the permutation π_p is unique, otherwise a class Π_p of several permutations is generated. The set of directions $\mathbf{p}$ at which π_p is not unique will be denoted $H(\mathbf{x}_1, \ldots, \mathbf{x}_n)$,

$$H(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \left\{\mathbf{p} \in S^{d-1} \,|\, \text{there are } i \neq j \text{ such that } \mathbf{p}'\mathbf{x}_i = \mathbf{p}'\mathbf{x}_j\right\}.$$

Consider weights $w_{j,\alpha}$ for $j \in \{1, 2, \ldots, n\}$ and $\alpha \in [0, 1]$ that satisfy the following restrictions (i) to (iii):

(i) $\sum_{j=1}^{n} w_{j,\alpha} = 1$, $w_{j,\alpha} \geq 0$ for all j and α,

(ii) $w_{j,\alpha}$ increases in j for all α,

(iii) if $\alpha < \beta$ then

$$\sum_{j=1}^{k} w_{j,\alpha} \leq \sum_{j=1}^{k} w_{j,\beta}\,, \quad k = 1, \ldots, n\,. \tag{3.2}$$

Then, as it has been shown in Dyckerhoff and Mosler (2011), the function $h_{D_\alpha(\mathbf{x}_1,\ldots,\mathbf{x}_n)}$,

$$h_{D_\alpha(\mathbf{x}_1,\ldots,\mathbf{x}_n)}(\mathbf{p}) = \sum_{j=1}^{n} w_{j,\alpha}\mathbf{p}'\mathbf{x}_{\pi_p(j)}\,,\ \mathbf{p} \in S^{d-1} \tag{3.3}$$

is the support function of a convex body $D_\alpha = D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, and $D_\alpha \subset D_\beta$ holds whenever $\alpha > \beta$.

Now we are ready to give the general definition of a family of weighted-mean trimmed regions.

Definition 3.1. (Dyckerhoff and Mosler) Given weights $w_{1,\alpha}, \ldots w_{n,\alpha}$ that satisfy the restrictions (i) to (iii), the convex body $D_\alpha = D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ having support function (3.3) is named the *weighted-mean α-trimmed region* of $\mathbf{x}_1, \ldots, \mathbf{x}_n$, $\alpha \in [0, 1]$.

The next proposition explains the name by stating that a weighted-mean trimmed region is the convex hull of weighted means of the data. Further it describes the region's extreme points.

Proposition 3.2. *It holds*

$$D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) = conv\left\{ \sum_{j=1}^{n} w_{j,\alpha}\mathbf{x}_{\pi(j)} \,\middle|\, \pi \text{ permutation of } \{1, \ldots, n\} \right\}, \tag{3.4}$$

and the set of extreme points of D_α is given by

$$Ext\big(D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)\big) = \left\{ \sum_{j=1}^{n} w_{j,\alpha}\mathbf{x}_{\pi_p(j)} \,\middle|\, \mathbf{p} \in S^{d-1} \setminus H(\mathbf{x}_1, \ldots, \mathbf{x}_n) \right\}. \tag{3.5}$$

Due to their attractive analytical properties, WM regions are useful statistical tools. Besides being *continuous* in the data and in α, they are *subadditive*, that is,

$$D_\alpha(\mathbf{x}_1 + \mathbf{y}_1, \ldots, \mathbf{x}_n + \mathbf{y}_n) \subset D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) \oplus D_\alpha(\mathbf{y}_1, \ldots, \mathbf{y}_n)\,,$$

and *monotone*: If $\mathbf{x}_i \leq \mathbf{y}_i$ holds for all i (in the componentwise ordering of $\mathbb{R}^d$), then

$$D_\alpha(\mathbf{y}_1, \ldots, \mathbf{y}_n) \subset D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) \oplus \mathbb{R}^d_+ , \quad \text{and}$$
$$D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) \subset D_\alpha(\mathbf{y}_1, \ldots, \mathbf{y}_n) \oplus \mathbb{R}^d_- ,$$

where $\oplus$ signifies the Minkowski sum of sets. For proofs and more results, like projection properties, the reader is again referred to Dyckerhoff and Mosler (2011).

3.2.2 Special notions of weighted-mean trimming

The general notion of WM regions provides a flexible approach to the trimming of multivariate data. Depending on the choice of the weights $w_{j,\alpha}$ different families of trimmed regions are obtained. They include the zonoid regions (Koshevoy and Mosler, 1997), the ECH and ECH* regions (Cascos, 2007), the geometrically trimmed regions, and many others. For an illustration in dimension $d = 3$ see Figure 3.1. Here the left panel shows zonoid regions for different parameters α, while the right one consists of ECH* regions for the same data and α. Note from Figure 3.1 that the surface of a zonoid region appears to have less facets than an ECH* region.

Historically, the first type of WM regions was *zonoid trimmed regions* $ZD_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ for $0 < \alpha \leq 1$ proposed by Koshevoy and Mosler (1997),

$$ZD_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \left\{ \sum_{i=1}^n \lambda_i \mathbf{x}_i \,|\, 0 \leq \lambda_i \leq \frac{1}{n\alpha}, \sum_{i=1}^n \lambda_i = 1 \right\} .$$

The corresponding support function is

$$h_{ZD_\alpha}(\mathbf{p}) = \sum_{j=1}^n w_{j,\alpha} \mathbf{p}' \mathbf{x}_{\pi_p(j)} ,$$

with weights

$$w_{j,\alpha} = \begin{cases} 0 & \text{if } j < n - \lfloor n\alpha \rfloor , \\ \frac{n\alpha - \lfloor n\alpha \rfloor}{n\alpha} & \text{if } j = n - \lfloor n\alpha \rfloor , \\ \frac{1}{n\alpha} & \text{if } j > n - \lfloor n\alpha \rfloor . \end{cases} \tag{3.6}$$

Many properties of the zonoid regions are developed in Mosler (2002); particularly important is that they contain full information about the data.

Another important notion of WM regions is that of ECH* regions (Cascos, 2007). Their support function

$$h_{ECH^*_\alpha}(\mathbf{p}) = \sum_{j=1}^{n} w_{j,\alpha}\, \mathbf{p}'\mathbf{x}_{\pi_p(j)}$$

employs the weights

$$w_{j,\alpha} = \frac{j^{1/\alpha} - (j-1)^{1/\alpha}}{n^{1/\alpha}}\,. \tag{3.7}$$

For a detailed discussion of these and other special weighted-mean trimmed regions, like ECH and geometrically trimmed regions, the reader is referred to Dyckerhoff and Mosler (2011).

3.3 Geometry of the algorithm

In this section we present the basic ideas of the algorithm. Specifically, it relies on notions from convex geometry.

3.3.1 Trimmed region as a convex polytope

Consider a data cloud, which is a finite set of data points, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^d$, and assume that the points are all different and in *general position* (i.e., no more than d of them lie on the same hyperplane).

For given α, the α-trimmed region $D_\alpha = D_\alpha(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ is a convex polytope in $\mathbb{R}^d$ that is bounded and closed. Such a polytope is the nonempty and bounded intersection of finitely many closed halfspaces. Thus the polytope can be completely described by its bounding hyperplanes. The intersection of a bounding hyperplane with the polytope is named a *facet* if it has the affine dimension $d-1$. Similarly, it is named a *ridge* if it has the dimension $d-2$. In dimension $d=3$ a facet is a *face*, and a ridge is an *edge*.

In the sequel, we calculate the weighted-mean trimmed regions by their facets. Two computational tasks will have to be repeatedly performed:

1. Calculate a facet,
2. find an adjacent facet.

A ridge is the intersection of two facets. Therefore, investigating the ridges is a way to extract information about the adjacency of facets. Each ridge of a given facet provides an indicator whether another facet is adjacent or not. A bounding hyperplane is fully

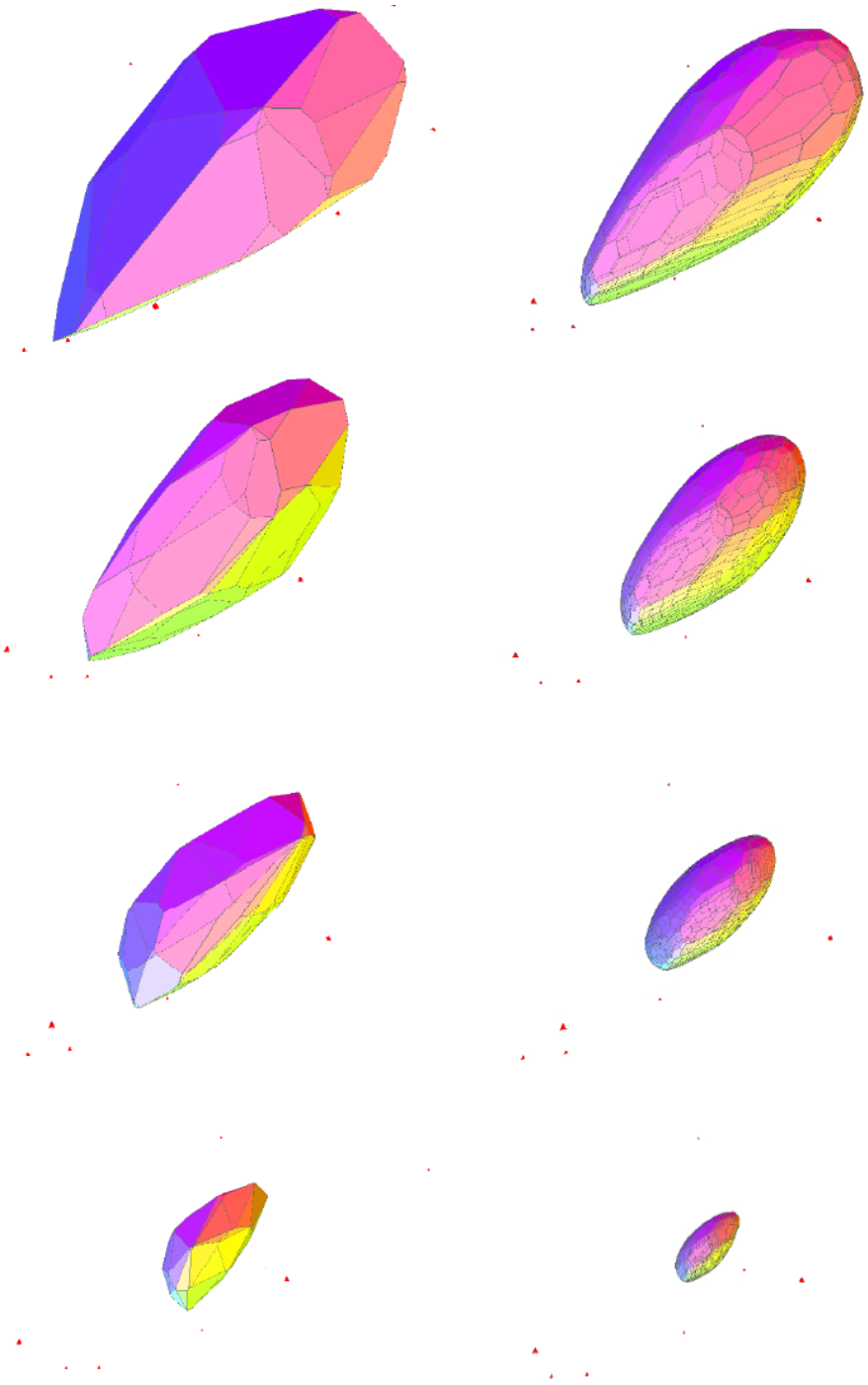

FIGURE 3.1: Examples of WM regions in $\mathbb{R}^3$. Representation of the zonoid (left) and ECH* (right) regions for the same data and depths.

described by its (outwards pointing) normal and one additional point, in particular one of its vertices. Hence, for every facet we determine its normal and a vertex as well as the adjacency indicator of each of its ridges. By doing this successively for all facets, a complete representation of the trimmed region is obtained.

Mosler, Lange, and Bazovkin (2009) develop an exact algorithm for calculating zonoid

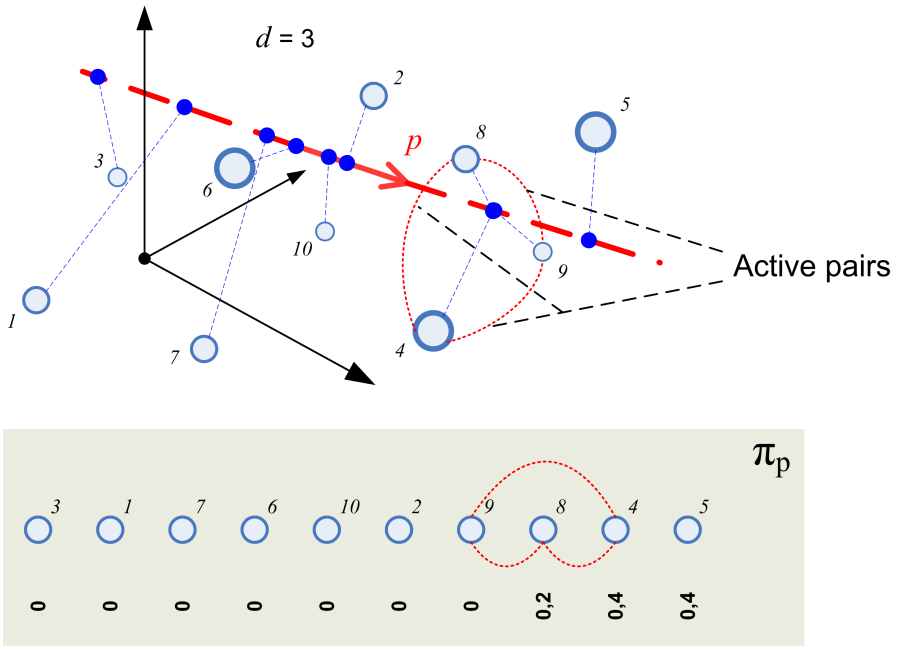

FIGURE 3.2: Characterizing the normal $\mathbf{p}$ of a facet (zonoid region, $d = 3, n = 10, \alpha = 0.25$): Data points and their projections; $\mathbf{p}$-ordered indices; weights; active pairs of indices.

trimmed regions. They demonstrate that, in the case of zonoid regions, the normal of a facet is characterized by d points of the data cloud.

Regarding a general WM region, we will firstly characterize its facets. Let F be a given facet of $D_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and $\mathbf{p}$ denote its normal. Then F has at least d vertices, which all are supported by $\mathbf{p}$. Due to (3.3) and (3.4) each vertex v has the form

$$v = \sum_{j=1}^{n} w_{j,\alpha} \mathbf{x}_{\pi_p(j)} \quad \text{with some} \quad \pi_p \in \Pi_p \,. \tag{3.8}$$

Consequently, not all $\mathbf{p}'\mathbf{x}_i$ can be different: It holds $\mathbf{p} \in H(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, and Π_p has at least d elements. Now let us consider the $\mathbf{p}$-ordered series of indices

$$\pi_p(1), \pi_p(2), \ldots, \pi_p(n) \,.$$

In the sequel we will mention those pairs of indices $(\pi_p(j), \pi_p(k))$ as *active* that satisfy the equation $\mathbf{p}'\mathbf{x}_{\pi_p(k)} = \mathbf{p}'\mathbf{x}_{\pi_p(j)}$ plus a restriction on their weights w_j and w_k, which will be specified below. The equation means that the difference $\mathbf{x}_{\pi_p(k)} - \mathbf{x}_{\pi_p(j)}$ is orthogonal to $\mathbf{p}$,

$$\mathbf{x}_{\pi_p(k)} - \mathbf{x}_{\pi_p(j)} \perp \mathbf{p} \,. \tag{3.9}$$

At a given $\mathbf{p}$, all indices that belong to an active pair will be mentioned as *active indices*, all others as *passive indices*.

From now on, we will distinguish *data points* and *data vectors*. By a data vector we mean the difference of two data points. To determine $\mathbf{p}$, $d-1$ data vectors are needed. Each of them is based on an active pair of indices and thus satisfies the orthogonality relation (3.9). As, by assumption, the data are in general position, any such $d-1$ data vectors are linearly independent. They will be mentioned as a *basis* of F and denoted by $\mathcal{V}_F$. Note that the basis of a facet is not unique: To form a basis, out of all active pairs of indices any $d-1$ pairs that yield linearly independent data vectors may be chosen. To summarize:

Theorem 3.3. (Basis of a facet) *The normal of a facet F is orthogonal to exactly $d-1$ linearly independent data vectors, which form a basis of F. The facet is characterized by a basis and one of its vertices.*

Next we develop the two essential steps of calculating a facet and finding an adjacent facet in detail.

Task 1: Calculating a facet In our algorithm we have to construct a basis for each facet of the polytope. Let $\mathbf{p}$ be the normal of a given facet F, choose some $\pi_p \in \Pi_p$, and consider the series of $\mathbf{p}$-ordered indices $\pi_p(1), \pi_p(2), \ldots, \pi_p(n)$. This series contains $d-1$ active pairs of indices, $\pi_p(j), \pi_p(k)$, that define a basis $\mathcal{V}_F$.

The special case of zonoid regions (having weights (3.6)) appears to be particularly simple: A facet is identified by exactly d data points (carrying serially $\mathbf{p}$-ordered indices), which yield $d-1$ linearly independent difference vectors that are orthogonal to $\mathbf{p}$ (Mosler et al., 2009). As an example, Figure 3.2 exhibits ten points in $\mathbb{R}^3$ and their projections to the line generated by $\mathbf{p}$. The lower panel contains the $\mathbf{p}$-ordered series of indices and the weights (3.6) for $\alpha = 0.25$. Here, three indices ($9, 8$, and 4) are active, as well as three index pairs ($(9,8), (9,4)$, and $(8,4)$). A basis of the facet is given, e.g., by the data vectors $\mathbf{x}_8 - \mathbf{x}_9$ and $\mathbf{x}_4 - \mathbf{x}_8$. Note that for these weights (at $\alpha = 0.25$) and *any* direction $\mathbf{p}$ the indices $\pi_p(7), \pi_p(8)$, and $\pi_p(9)$ become the active ones.

Other types of weighted-mean trimmed regions employ less simple weights. With them the number of active indices involved in the identification of a facet F may be larger than d. E.g., Figure 3.3 illustrates the characterization of a facet of an ECH* region, with weights (3.7) and $\alpha = 0.25$. It shows another example of ten points in $\mathbb{R}^3$ and their projections, given some $\mathbf{p}$. In this example, four indices ($7, 6, 4$, and 2), and two index pairs ($(7,6)$ and $(4,2)$) are active, and a basis consists of $\mathbf{x}_6 - \mathbf{x}_9$ and $\mathbf{x}_2 - \mathbf{x}_4$, being unique up to sign.

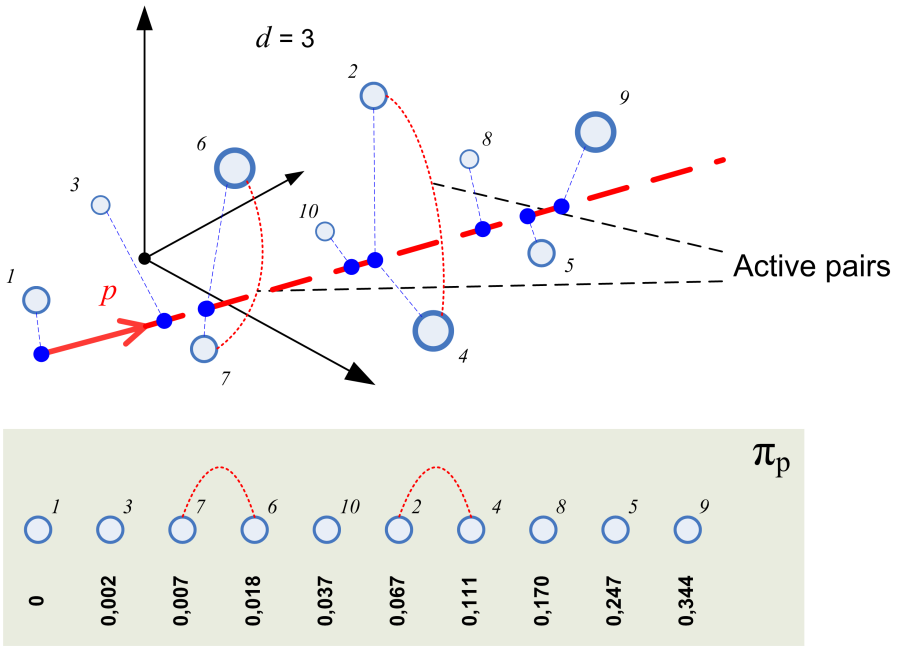

FIGURE 3.3: Characterizing the normal $\mathbf{p}$ of a facet (ECH* region, $d = 3, n = 10, \alpha = 0.25$): Data points and their projections; $\mathbf{p}$-ordered indices; weights; active pairs of indices.

In general, we consider the following disjoint blocks $\mathcal{A}_l$ of indices, $l = 1, \ldots, L$,

$$\mathcal{A}_l = \{\pi_p(i) \,|\, i \in \{a_l, a_l + 1, \ldots, a_l + n_l - 1\},\ \mathbf{p}'\mathbf{x}_{\pi_p(i-1)} = \mathbf{p}'\mathbf{x}_{\pi_p(i)} \text{ for } i > a_l\}\,,$$

where $a_{l-1} < a_l$ holds ($a_0 = 0$), and define: A pair of indices is called *active* if a block $\mathcal{A}_l$ exists that contains both of them. In particular, each block contains at least two elements, $n_l \geq 2$, and it holds $w_{a_l,\alpha} < w_{a_l+n_l-1,\alpha}$, which is the restriction on weights announced above. Moreover, $\mathcal{A}_l \cap \mathcal{A}_m = \emptyset$ if $l \neq m$, and

$$\mathcal{V}_F = \bigcup_{l=1}^{L} \{\mathbf{x}_{\pi_p(i)} - \mathbf{x}_{\pi_p(i+1)} \,|\, \pi_p(i), \pi_p(i+1) \in \mathcal{A}_l\}\,.$$

Note that in the case of zonoid regions only one block of active indices arises; it holds $L = 1$.

The remaining indices, which are not in $\bigcup_{\ell=1}^{L} \mathcal{A}_l$, are the *passive* ones. Among them we distinguish disjoint blocks that have equal weights,

$$\mathcal{B}_k = \{\pi_p(i) \,|\, i \in \{b_k, b_k + 1, \ldots, b_k + m_k - 1\},\ w_{i-1,\alpha} = w_{i,\alpha} \text{ for } i > b_k\}\,,$$

$k = 1, 2, \ldots, K$, while $m_k \geq 1$, $b_{k-1} < b_k$ with $b_0 = 0$, and $w_{b_{k-1},\alpha} < w_{b_k,\alpha}$.

Thus $\pi_p(1), \pi_p(2), \ldots, \pi_p(n)$ divides into a series $\mathcal{S}_F$ of blocks $\mathcal{A}_l$ and $\mathcal{B}_k$ of active and passive indices, respectively. Observe that these blocks may occur in any order, which

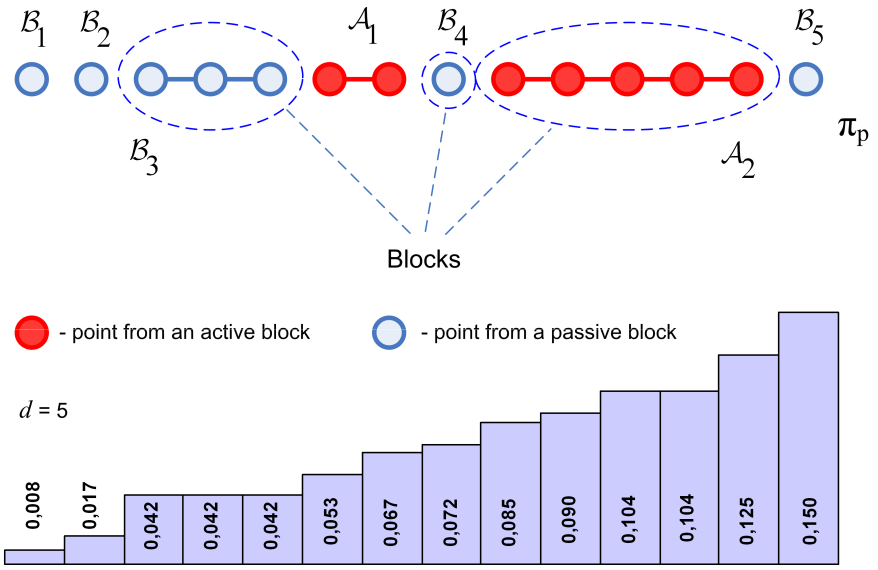

FIGURE 3.4: Series of blocks of active and passive indices; weights as indicated.

is illustrated in Figure 3.4.

Task 2: Finding an adjacent facet To identify adjacent facets we start from a given facet F, which has support vector $\mathbf{p}$ and which from now on will be called the *current facet*. Each ridge of F offers a way of "jumping" to a neighboring facet. Therefore we investigate the ridges of the current facet F and, consequently, its adjacent facets. Each element of the basis $\mathcal{V}_F$ may be regarded as a reduction of one degree of freedom of the support vector $\mathbf{p}$. To determine $\mathbf{p}$ as the normal of the current facet F, we have to reduce $d-1$ degrees of freedom and calculate the uniquely determined support vector $\mathbf{p}$ that is orthogonal to $d-1$ linearly independent data vectors (differences of vectors from the original data cloud). A ridge of the current facet is supported by vectors that result from adding one degree of freedom to the given support vector $\mathbf{p}$. The degree of freedom is added by leaving out one of the $d-1$ data vectors from the basis $\mathcal{V}_F$, or, more generally, by replacing some k data vectors in $\mathcal{V}_F$ with some $k-1$ ones, while keeping linear independence within the basis.

Removing one element from the basis $\mathcal{V}_F$ corresponds to splitting one of the active blocks in $\mathcal{S}_F$, say $\mathcal{A}_l$, into $\mathcal{A}_l^1$ and $\mathcal{A}_l^2$. By this, a modified series of blocks, $\mathcal{S}_{F*}$, is obtained. Observe that, if $\mathcal{A}_l^1$ (or $\mathcal{A}_l^2$) is a singleton, its element becomes a passive index in $\mathcal{S}_{F*}$.

Now, the removed element of the basis has to be substituted by another data vector. For this, any pair (i^*, j^*) of indices that belong to two neighboring blocks of $\mathcal{S}_{F*}$ can be chosen and the corresponding data vector $\mathbf{x}_{i^*} - \mathbf{x}_{j^*}$ be added to the basis. (However, no

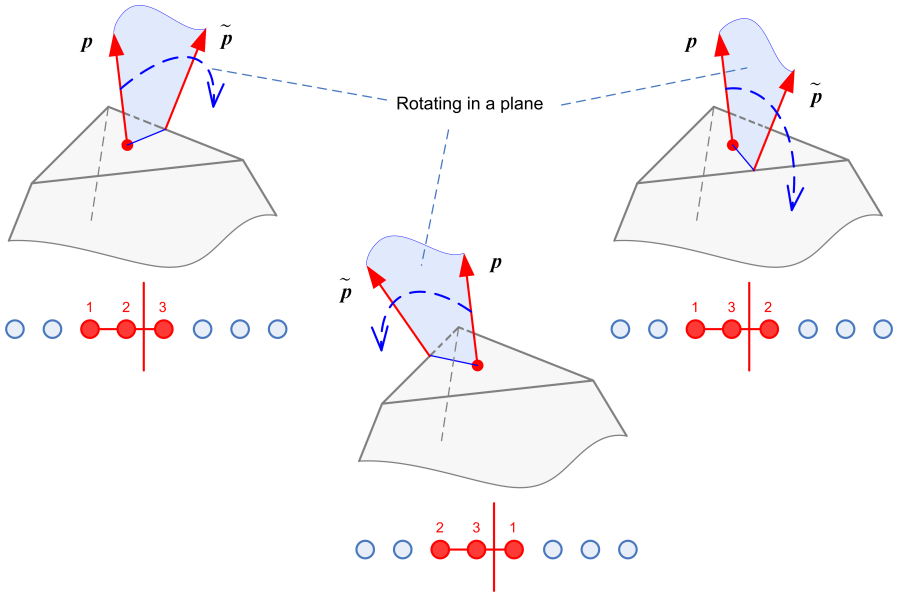

FIGURE 3.5: Rotating $\mathbf{p}$ in a plane of dimension two in $\mathbb{R}^d$.

pair from $\mathcal{A}_l^1 \times \mathcal{A}_l^2$ must be selected.) Then the new basis defines a facet that is adjacent to the current facet F.

This step may be visualized as follows (see Figure 3.5 for $d = 3$): Starting at $\mathbf{p}$, the support vector is rotated in a plane (of dimension two in $\mathbb{R}^d$) until another data vector enters the basis $\mathcal{V}_F$, i.e., until another data vector becomes orthogonal to $\mathbf{p}$. Let $\mathcal{E}_p$ denote the set of vertices of the polytope corresponding to the support vector $\mathbf{p}$. We turn $\mathbf{p}$ until it stops at the position $\tilde{\mathbf{p}}$ where $\tilde{\mathbf{p}}'\mathbf{x}_{i^*} = \tilde{\mathbf{p}}'\mathbf{x}_{j^*}$ for some i^* and j^*, i.e., $(\mathbf{x}_{i^*} - \mathbf{x}_{j^*}) \perp \tilde{\mathbf{p}}$. Then, if $\mathcal{E}_{\tilde{\mathbf{p}}} \supset \mathcal{E}_p$, this means that $\tilde{\mathbf{p}}$ is a normal to some facet $\tilde{F}$ which is a neighbor to the current facet. Otherwise, $\mathbf{p}$ is turned further until the condition is met. Obviously, to meet the condition, the indices i^* and j^* must be in different blocks of $\mathcal{S}_{F*}$. On the other hand, indices can continuously interchange places only with their neighbors, that is, $\mathbf{x}_{i^*}$ and $\mathbf{x}_{j^*}$ must be in blocks that neighbor each other.

So far we have exchanged a single basis vector against another one. However, the elements within each active block at $\mathbf{p}$ can be arbitrarily rearranged, and each active index used in the exchange step just *represents* a class of equivalent active indices. Therefore more than one, say k, active pairs living on $\mathcal{A}_l^1 \times \mathcal{A}_l^2$ may be exchanged simultaneously.

As a result of the basis exchange we have found a single adjacent facet. Our next task is to identify *all* facets that are adjacent to the current facet. For this, it is not necessary to enumerate all pairs of indices from neighboring blocks of $\mathcal{S}_{F*}$. Note that the elements of each active block $\mathcal{A}_l$ are equivalent in the $\mathbf{p}$-order, i.e., $\mathbf{p}'\mathbf{x}_{i^*} = \mathbf{p}'\mathbf{x}_{j^*}$ for all $i^*, j^* \in \mathcal{A}_l$. Hence, we may permute indices *within the active blocks* in an arbitrary way, which

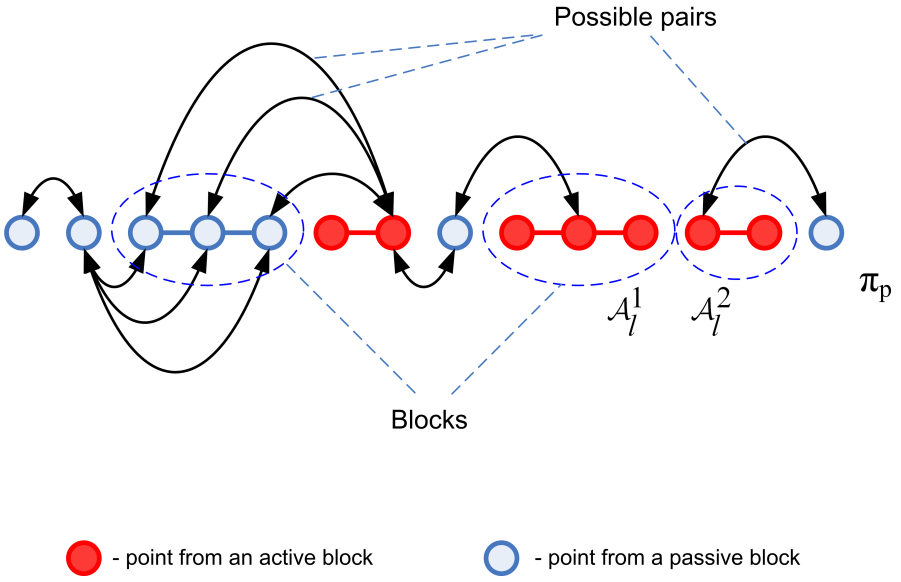

FIGURE 3.6: The series $\mathcal{S}_{F*}$ of blocks; with possible critical pairs.

means employing some other permutation from Π_p in place of the given permutation π_p. Therefore, in generating all possible basis exchanges, we need not consider all active indices for pairing, but may restrict to a *representative index* of each active block, say $r_l \in \mathcal{A}_l$, $l = 1, \ldots, L$. However, in the passive blocks, all indices have to be taken into account.

A pair (i^*, j^*) from two neighboring blocks in $\mathcal{S}_{F*}$ is called a *critical pair* if it consists of indices that are either passive or representative active indices. More formally, we may write the series $\mathcal{S}_{F*}$ of active and passive blocks as

$$\mathcal{S}_{F*} = (\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{L+K})$$

and define

$$\tilde{\mathcal{C}}_m = \begin{cases} \{r_l\} & \text{if } \mathcal{C}_m = \mathcal{A}_l \text{ for some } l, \\ \mathcal{B}_k & \text{if } \mathcal{C}_m = \mathcal{B}_k \text{ for some } k. \end{cases}$$

Then the set of critical pairs (that have to be checked for finding all adjacent facets) is given by

$$\bigcup_{m=1}^{L+M-1} \tilde{\mathcal{C}}_m \times \tilde{\mathcal{C}}_{m+1}\,. \tag{3.10}$$

The two computational tasks, calculating a facet and finding a neighboring facet, are performed until all facets of the polytope have been visited and computed. As a result of the algorithm, the WM region is completely described by its facets. Alternatively and in addition, we may be interested in calculating vertices of the polytope. These are easily determined by the following procedure.

Proposition 3.4. (Calculating vertices) *Consider a facet F having normal $\mathbf{p}$. Each vertex of F exactly corresponds to a permutation of $(\pi_p(1), \dots \pi_p(n))$ that is restricted to permutations within the $\mathcal{A}_l$.*

Corollary 3.5. *The minimum possible number of vertices of a facet is d (e.g., for zonoid regions). The maximum possible number of vertices of a facet is $d!$.*

E.g., in the case of ECH*-regions, the number of vertices of a facet varies from d to $d!$.

3.3.2 Spanning tree order

Based on the adjacency information obtained by the above approach we are able to calculate the facets in a sequential order. For this sequence, we use the spanning tree order (STO) discussed in Mosler, Lange, and Bazovkin (2009). The STO provides a complete ordering of all facets according to which they are generated in the algorithm. The general idea is:

1. Represent all facets adjacency information by a tree,
2. organize an efficient procedure to traverse the tree.

In the algorithm we apply a breadth-first search like that described in Knuth (1997). Using the STO we generate each facet only once, which is an efficient procedure.

Moreover, as the STO is based on the neighboring relation among facets, we can restrict the calculation of facets to some connected part of the trimmed region's surface, e.g. the part having support vector $\mathbf{p} \geq \mathbf{0}$. This proves to be useful in certain applications like multivariate risk measurement.

Note that we calculate the trimmed region by sequentially generating its facets, but not its vertices. In dimension $d = 2$ it is also possible to determine the region by enumerating its vertices; this is done by means of a so called *circular sequence* (Edelsbrunner, 1987).

It is easy to see that the proposed procedure applies to any choice of a weighting function satisfying the above WM restrictions (i) to (iii). Thus the algorithm is able to calculate any weighted-mean trimmed region.

Finally, we would like to turn the reader's attention again to the the adjacency of the sequentially generated vertices. That is a practical advantage because we can restrict the calculation to some specified part of the WM region which we might only be interested in. In this respect our procedure reminds of the so-called "gift-wrapping" approach,

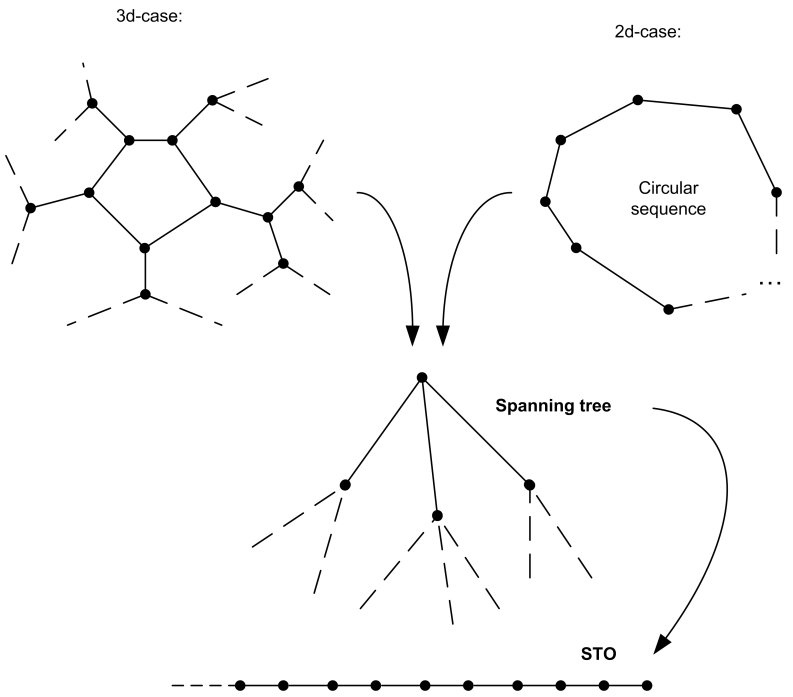

FIGURE 3.7: The sample scheme of the procedure.

which is used to solve common tasks of constructing convex polytopes, in particular calculating the convex hull of a given set, where algorithms for higher dimensions have been proposed by Swart (1985) and others.

However, the structure of a WM region is much more complex, since it aggregates not only local information, as it is the case in the construction of a convex hull, but depends on information on the whole data cloud, including all inner points. If $0 \leq \alpha \leq \frac{1}{n}$ the WM region turns into a convex hull of data points, which is the trivial case in our task. For this reason our algorithm differs fundamentally from a classical "gift-wrapping" procedure. Other than Swart's and similar approaches we find a facet of the WM region only once (in contrast to a repeated finding of facets and removing the new one after discovering the duplication), that is, we make no redundant calculations and form a unique chain of facets according to the STO. Furthermore, convex hull algorithms work with a given set of points, while in our problem there is no such set in an explicit form, and facets are constructed without having information on their vertices. Besides this, it was shown above that a facet of a WM region is, in most cases, no $(d-1)$-dimensional simplex. Actually, the number of vertices can blow up to $d!$, which represents a difficult case for a convex hull algorithm.

3.4 The algorithm

3.4.1 Interface and steps

In this subsection we give a formal scheme of the algorithm and an interface to it.

Input

d (dimension of the data space, $d \geq 2$);

n (number of data points, $n > d$);

`cloud` (data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$);

α (depth parameter);

w_α (weight vector; alternatively: name of special type of WM regions).

Output

- `trimmed_region` (all facets of the trimmed region, with coordinates of their vertices);
- *Visualization.*

Steps of the Algorithm

A. *Initialization: Read the input.*

B. *Determine a first facet:*

   a. From `cloud`, *form a set* `vec_defining_set` of $d-1$ linearly independent data vectors (= basis).

   b. *Calculate*, to the hyperplane through `vec_defining_set`, a normal vector $\mathbf{r}$.

   c. *Substitute* $\mathbf{r}$ for $\mathbf{p}$ and choose a permutation $\pi_p \in \Pi_p$. *Determine* the series of active blocks $\{\mathcal{A}_l\}_{l=1\ldots L}$ in this permutation.

   d. $\{\mathcal{A}_l\}_{l=1\ldots L}$ defines `vec_defining_set` and, hence, the first facet `ffacet`.

   e. *Place* `ffacet` $\hookrightarrow$ the head of `queue`.

Having the initial facet, we can start a sequential calculation of all others.

C. *Determine all facets:*

a. *Take* `curr_facet` $\hookleftarrow$ front of the `queue`.

b. *Create* neighboring facets of `curr_facet`.

I. *Create all ridges* by adding a degree of freedom to $\mathbf{p}$ (reducing cardinality of the basis `vec_defining_set` by one).

i. Take the next $\mathcal{A}_l$ and create all possible splittings of it into two subsets: $\langle \mathcal{A}_l^1, \mathcal{A}_l^2 \rangle$. *Replace* $\{\mathcal{A}_l\}$ by $\mathcal{A}_l^1$ and $\mathcal{A}_l^2$. *If* either $\mathcal{A}_l^1$ or $\mathcal{A}_l^2$ is a singleton, *remove it* from the active blocks. A set `partial_facets(l)` is obtained.

ii. *Drop off* all elements of `partial_facets(l)` that are no active blocks. A set `ridges(l)` is obtained.

iii. *Add* `partial_facets(l)` to the set `ridges`. *If* an unprocessed $\mathcal{A}_l$ is left, go to C.(b.)I.i.

Now we have found all ridges and are ready to do "jumps" to neighboring facets. Note that the procedure jumps *only to new facets*. In doing this, we process each facet twice: first, we only preprocess it by marking its ridges; second, we do a normal calculation of the "jumps".

II. For the next ridge in `ridges` do the following:

i. *If* `curr_facet` is not `preprocessed`, *calculate a hash code* of the ridge and mark it $\hookrightarrow$ `hash_table`. Then go to C.(b.)II.

ii. *Check* in `hash_table`, whether the ridge *is blocked*. *If* yes, go to C.(b.)II.

iii. *Build* the maximum number of linearly independent data vectors that are based on active pairs. *Put* the vectors as rows into a matrix A. There will be $d-2$ rows.

iv. Given a normal vector $\mathbf{r}$ to `curr_facet`, *put it* as an additional row into A. *Put* any non-zero vector that is linearly independent of the $d-1$ previously chosen rows as a last row into A. Let b be a vector that consists of $d-1$ zeros and a last non-zero entry.

v. *Solve* the linear equation $A\,z = b$. Its solution z and r span a plane B_2 that is orthogonal to the ridge.

vi. *Calculate critical pairs* according to (3.10).

vii. *Rotate* $\mathbf{p}$ in the plane B_2. In doing so, start at $\mathbf{p} = \mathbf{r}$ and *move* $\mathbf{p}$ in a way that the new ordering of points in the permutation corresponds to the previous splitting of an active block.

viii. *Stop if* $\mathbf{p}$ becomes orthogonal to some vector built on a critical pair of indices. *Take this vector* as `new_vector`.

The neighboring facet is discovered. Now we have to reconstruct its combinatorial stucture.

ix. *Add* `new_vector` to `vec_defining_set`. `new_facet` is obtained. The current position of $\mathbf{p}$ is a normal $\mathbf{r}$ to `new_facet`.
If `new_vector` is built on indices from an active block $\mathcal{A}_j$ and a neighboring passive block, then *augment* $\mathcal{A}_j$ with the passive index.
If `new_vector` is built on indices from two active blocks, $\mathcal{A}_j$ and $\mathcal{A}_{j+1}$, then *merge* these two blocks.
If `new_vector` is built on two passive indices, then a new block $\mathcal{A}'_j$ is created having them as its two elements.

x. *Place* `new_facet` $\hookrightarrow$ the head of `queue`.

We have to mark the ridges of the facet immediately, thus preventing another "jump" to the facet. The immediate processing is enabled by putting the new facet to the head of the queue.

III. *If* `curr_facet` is not `preprocessed`, *label it* as `preprocessed` and *place* $\hookrightarrow$ `queue`. Then, go to C.a.

IV. For `curr_facet`, *calculate* the vertices and its absolute distance from the origin by (3.8).

V. *Shift* `curr_facet` by $\overline{\mathbf{x}}$ and *transfer it* to `trimmed_region`.

c. *If* `queue` is not empty, go to C.a. Otherwise, *stop*: Then, `trimmed_region` contains all facets of the trimmed region.

We would also direct a reader's attention to three special features of the algorithm:

1. Using a *"double-hash"*: The ridges are hashed using a *"double-hash"* table. That is, a ridge is `blocked` if it has been marked twice.

2. Calculating the hash code: The hash code is calculated by creating a bit row from integer numbers describing $\{\mathcal{A}_l\}_{l=1\ldots L}$ and one number describing the absolute position of a ridge (to distinguish parallel ridges).

3. Determining all adjacent facets: For optimizing the complexity at this stage the mutual information concerning all ridges can be used. The details of such a heuristic are described in Section 3.8.

3.4.2 Complexity of the algorithm

Due to the mechanism of the *"double-hash"* the algorithm has as many loops as the WM region has facets. Obviously, this is the minimum number of facet generating loops in this sort of algorithms.

At each facet F we have to calculate the normal of the facet and its distance from the origin. Further we have to determine all neighboring facets. This is done by solving linear equations and calculating inner products only. We have shown above that the complexity of this algorithmic loop amounts to $\mathcal{O}(d^2 \cdot R(F))$, where $R(F)$ is the number of ridges of the facet. $R(F)$ can vary between d and 2^{d-1}, depending on the type of the WM region and on α.

To obtain a rough conservative estimate of $R(F)$, we may proceed as follows: First, note that $R(F)$ is bounded by $\bar{R} = \prod_{l=1}^{L} 2^{n_l - 1}$. Then suppose that the active blocks $\mathcal{A}_l$ have about equal size and that their number L, as a first approximation, is proportional to the dimension d, say $L \approx d/c$. Under these assumptions $\bar{R} \approx L \cdot 2^{d/L-1} \approx d/c \cdot 2^{c-1}$, that is, $R(F)$ is approximately bounded by the dimension d multiplied with a constant $K = 2^{c-1}/c$ that does not depend on the dimension.

Searching for all neighbors of a facet, we have to calculate n inner products, which gives complexity $\mathcal{O}(nd)$. Hence, the complexity of one facet generating loop is described by $\mathcal{O}((d^2 + nd) \cdot \bar{R}) \cong \mathcal{O}(d^2 n \cdot K)$, since $n > d$. If the average number of facets is denoted by $N(n, d)$, the average computational complexity of the algorithm amounts to $\mathcal{O}((d^2 + nd) \cdot \bar{R} \cdot N(n, d)) \cong \mathcal{O}(d^2 n \cdot K \cdot N(n, d))$.

For a better understanding of $N(n, d)$ we like to discuss the number of vertices $V(n, d)$ of the WM region. It is maximal when all weights in the weight vector are distinct. Let us consider hyperplanes that are orthogonal to all data vectors and intersect at the origin. Then the $\mathbb{R}^d$ is split by the hyperplanes into convex cones, and there will be a bijection between the vertices of the WM region and these convex cones[1]. Winder (1966) has shown that the number of such cones equals $2 \sum_{i=0}^{d-1} \binom{m-1}{i}$ for m hyperplanes, which is $\mathcal{O}(m^d)$. We have at most $\frac{n(n-1)}{2}$ hyperplanes (for zonoid regions this bound reduces to $\mathcal{O}(n)$) and, therefore, obtain an $\mathcal{O}(\frac{n^{2d}}{2^d})$ upper bound for $V(n, d)$. It means that $V(n, d)$ lies between $\mathcal{O}(n^d)$ and $\mathcal{O}(\frac{n^{2d}}{2^d})$ depending on the weight vector. In turn, we have already seen, that each facet contains up to $d!$ vertices, which leads to $N(n, d) \ll V(n, d)$.

Regarding the hash table of created facets, each facet occupies $\mathcal{O}(d \cdot \log_2 n)$ storage size, while the hash table, in almost any case, has a constant size C, not depending on n and d. Therefore, the use of general memory is of the order $\mathcal{O}(N(n, d) \cdot d \cdot \log_2 n + C)$. Facets, once they have been created, are put into a secondary store, thus considerably lowering the storage cost.

Table 3.1 exhibits the results of a first small simulation study. It gives an idea how the time for computing a single facet depends on d and n and how it varies with several types of WM regions: zonoid, ECH*, ECH, and geometrically trimmed regions (for the

[1] Cf. *direction domains* in Mosler, Lange, and Bazovkin (2009).

WMTD type	d	n	Time per facet	Total time [sec]
Zonoid	3	10	0.009700	0.445
Zonoid	3	15	0.013840	1.531
Zonoid	4	10	0.012474	1.609
Zonoid	4	15	0.015862	14.140
Zonoid	5	10	0.017370	2.398
Zonoid	5	15	0.022335	40.953
ECH	3	10	0.009111	0.843
ECH	3	15	0.012212	2.375
ECH	4	10	0.015255	21.891
ECH	4	15	0.019632	97.765
ECH	5	10	0.023519	117.625
ECH	5	15	0.029733	1032.75
ECH*	3	10	0.009610	0.750
ECH*	3	15	0.012218	1.617
ECH*	4	10	0.015286	22.922
ECH*	4	15	0.020011	94.070
ECH*	5	10	0.022970	139.070
ECH*	5	15	0.029660	890.68
Geometrical	3	10	0.009355	0.930
Geometrical	3	15	0.013056	1.101
Geometrical	4	10	0.015356	23.805
Geometrical	4	15	0.020157	93.406
Geometrical	5	10	0.023036	137.312
Geometrical	5	15	0.029794	1028.51

TABLE 3.1: Sample computational results of the WM regions algorithm.

latter two, see Dyckerhoff and Mosler (2011)). The data is distributed uniformly on a d-dimensional cube. We focus on the time per facet (TpF) because it characterizes the efficiency of the algorithm in a most obvious way. The total computational time amounts to the latter multiplied by the number of facets, which is a parameter depending only on the data. We observe that the TpF shows the following growth behavior: Approximately linear on n and slightly convex on d, which may be seen as some low order polynomial dependency on dimension.

3.5 The *R* package WMTregions

The algorithm has been programmed as an R package and named WMTregions (Bazovkin and Mosler, 2011). It is available for downloading from Comprehensive R Archive Network at `http://CRAN.R-project.org/package=WMTregions`. Properly, the main functionality has been realized in $C++$ and the R part is used as (i) a thin client for the pre-compiled routine, (ii) the user interface and (iii) for the visualization. In the next

subsection we consider it in detail. The formal description of the functions and architecture will be followed by two examples of applying the package to simulated and to real data.

3.5.1 Technical overview

Dependencies An autonomously compiled *C++* program provides a $3d$-visualization as it is shown, for instance, in Figure 3.8. The visualization is designed by means of a cross-platform graphical specification *OpenGL*. In turn, in the *R* package we access the *OpenGL* functionality by means of the rgl package (Adler and Murdoch, 2011).

The less powerful but applicable for the data of any dimension, vertices based visualizing is realized with the help of the rggobi package (Lang et al., 2010). The latter also uses graphical toolkit *GTK+* through its *R* proxy package RGtk2 (Lawrence and Lang, 2010). To be able to use it you must first install *GTK+* library (`http://www.gtk.org/download`) on your machine. On the most systems this installation is proposed automatically while getting RGtk2. If not, you must do it manually before using the package. Moreover, the old versions of *GTK+* and *ggobi* can cause problems in installing and using RGtk2 and rggobi: If the packages fail, you must reinstall *GTK+* and *ggobi*.

***R* functions** The package contains functions for calculating and representing WM regions:

- Function `WMTR(fname = "Cloud.dat", fdir = getwd(), bound = 0)`.

  Goal: Calculates the WM region.

  Arguments:

  - `fname`: the name of the data input file (see Subsection 3.5.1) in the directory `fdir`.
  - `fdir`: a path to the directory where the input and output files will be located. The default value is the *R* working directory.
  - `bound`: an option of additional outputting the lower or the upper boundary of the WM region (i.e., $\partial\left(D_\alpha(\mathbf{x}_1,\ldots,\mathbf{x}_n)\oplus\mathbb{R}^d_+\right)\cap D_\alpha(\mathbf{x}_1,\ldots,\mathbf{x}_n)$ or $\partial\left(D_\alpha(\mathbf{x}_1,\ldots,\mathbf{x}_n)\oplus\mathbb{R}^d_-\right)\cap D_\alpha(\mathbf{x}_1,\ldots,\mathbf{x}_n)$, respectively). `-1` corresponds to the lower one; `1` - to the upper one; `0`, the default value, makes no additional output.

  Output:

  - A file `"TRegion.dat"` in the directory `fdir`. The calculated WM region with facets represented by their normals and intercepts.
  - A file `"TRegion_vertices.dat"` in the directory `fdir`. The calculated WM region with facets represented by the coordinates of their vertices.
  - Auxiliary files `"TRegion_bound.dat"` and `"TRegion_vertices_bound.dat"` with a bound of the calculated WM region.

  Description: This function is the main function, which reads input data from an input file `fname` and writes the result into an output file "TRegion.dat", both files being located in `fdir`. The format of the files is described below in Subsection 3.5.1.

- Function `visualWMTR(fdir = getwd())`.

  Goal: Visualizes the calculated WM region for the data in $\mathbb{R}^3$.

  Arguments:

  - `fdir`: a path to the directory where the output files of `WMTR()` are located. The default value is the R working directory.

  Output: Void value. The visualization of the calculated WM region and the data cloud points in a separate window under R environment.

  Description: This function realizes the $3d$-visualization of the data based on the computational results of the `WMTR()` function. The parameter `fdir` must be the same as was used in `WMTR()`.

- Function `showWMTR(fdir = getwd())`.

  Goal: Exhibits the calculated WM region of any dimension by making multiple projections of its vertices into $\mathbb{R}^3$.

  Arguments:

  - `fdir`: a path to the directory where the output files of `WMTR()` are located. The default value is the R working directory. Must be the same `fdir` as in `WMTR()`.

  Output: Void value. The rggobi visualization of the calculated WM region (represented only by its vertices) in separate windows under R environment.

  Description: The function visualizes a calculated WM region as a convex polytope by representing its vertices in rggobi (Lang et al., 2010) interactive graphics framework. The visualization is a series of projections into $\mathbb{R}^3$. The whole interaction toolset of the rggobi package, such as "$2d$ tour" or the projection pursuit, can be used here. In comparison with `visualWMTR()`, `showWMTR()` visualizes only vertices but, however, does it for the data of any dimension.

- Function `loadWMTR(fname = "TRegion.dat", fdir = getwd())`.

  Goal: Loads the calculated WM region of $d =$ `dim` into a matrix object.

  Arguments:

  - `fname`: the name of the file that contains the calculated WM region (the normal-intercept representation output file of `WMTR()`) in the directory `fdir`. The default name is "TRegion.dat".
  - `fdir`: a path to the directory where the file `fname` is located. The default value is the R working directory.

  Output:

  - A matrix object containing the normal-intercept coordinates of the WM region facets as its rows.

  Description: This function loads the calculated WM region of $d =$ `dim` into a matrix object in order to work with it as with a variable in R, for example, in using the function `pointinTR()`.

- Function `pointinTR(dpoint, tregion)`.

  Goal: Checks whether a point is in a specified trimmed region.

  Arguments:

  - `dpoint`: a vector containing the coordinates of the point to be checked.
  - `tregion`: a matrix object containing the WM region in the normal-intercept representation.

  Output:

  - Boolean value, whether `dpoint` is contained by `tregion`.

  Details: `tregion` is normally produced by the `loadWMTR()` function basing on a calculated WM region.

- Function `generTRsample(fname, fdir, dim, num, alpha, trtype)`.

  Goal: Generates sample data cloud file in a format appropriate for applying `WMTR()`.

  Arguments:

  - `fname`: the name of the output file.
  - `fdir`: a path to the directory where the file `fname` should be located.
  - `dim`: the dimension d of the data cloud.

– `num`: the number of points in the data cloud.

– `alpha`: the depth parameter.

– `trtype`: the notion of the WM region to be calculated.

Output:

– A file `fname` in the directory `fdir` with a data cloud of the specified parameters.

Description: This function is an auxiliary one. It generates a random uniformly distributed data cloud of any size `num` and any dimension `dim` with a format of an input file described in Subsection 3.5.1. With the default values of its arguments it looks as follows: `generTRsample(fname = "Cloud.dat", fdir = getwd(), dim = 3, num = 20, alpha = 0.05, trtype = "zonoid")`.

Input and output In this subsection we describe the format of the input and output information, which is represented by input and output files.

1. The input file. A data cloud is read from a text file of the following format (the sequence is fixed):

   - Type of the trimmed region (zonoid, ECH, ECH*, geometrically trimmed; given weight vector)
     Format: A text value from the following set: `"zonoid"`, `"ECH"`, `"ECH*"`, `"geometrical"`, `"general"`. `"general"` is used for the case when the weights are given manually instead of being automatically generated basing on the WM region type and the depth parameter.
   - Depth parameter
     Format: A floating point number from the interval $[0, \ldots, 1)$.
   - Dimension
     Format: An integer number $d \geq 2$.
   - Number of points of the data cloud
     Format: An integer number $n > d$.
   - *(If the type* `"general"` *is selected)* The weight vector
     Format: n non-decreasing floating point numbers matching the requirements for the weight vector.
   - Coordinates of each point
     Format: n groups of d floating point numbers, each group representing the coordinates of a point from the data cloud.

The points must be in the general position.

For example, to calculate trimmed regions of a data cloud made of 7 points we have to input the following, where the left column refers to a zonoid region with depth parameter 0.05, and the right column to general WM region defined by the weight vector `(0.02 0.02 0.03 0.15 0.15 0.26 0.37)`:

```
zonoid                                  general
0.05                                    0.00
3                                       3
7                                       7
3.433465 3.67261 2.985222               0.02 0.02 0.03 0.15 0.15 0.26 0.37
0.6119484 7.996853 6.70113              3.433465 3.67261 2.985222
6.429673 9.318805 5.684797              0.6119484 7.996853 6.70113
4.094673 3.255387 0.7768149             6.429673 9.318805 5.684797
4.764675 7.401488 1.766797              4.094673 3.255387 0.7768149
3.571828 4.102897 2.325751              4.764675 7.401488 1.766797
1.063743 0.7078045 8.968969             3.571828 4.102897 2.325751
                                        1.063743 0.7078045 8.968969
```

A sample input file with any given parameters and random by generated coordinates is provided by the function `generTRsample()`.

2. Output files.

   The whole calculated WM region is represented twofold:

   - `"TRegion.dat"`.

     An output file `"TRegion.dat"` consists of lines, each representing a facet of the trimmed region in the normal-intercept format, namely by $d+1$ numbers giving the equation of the hyperplane containing the facet. The first d of these numbers are coordinates of a normal to the facet, which is directed outward the WM region. The last number defines the intercept. For example,

     ```
     0.54301 0.43048 0.72100 -13.488
     ```

     corresponds to the hyperplane $\{x \in \mathbb{R}^3 : (0.54301\ 0.43048\ 0.72100) \cdot x - 13.488 = 0\}$.

   - `"TRegion_vertices.dat"`.

     An output file `"TRegion_vertices.dat"` consists of lines, each representing a facet of the trimmed region by the coordinates of its vertices. The coordinates of vertices are given in parentheses, while the vertices of a facet are again collected in parentheses. For example, a single facet ($d = 3$) is given by:

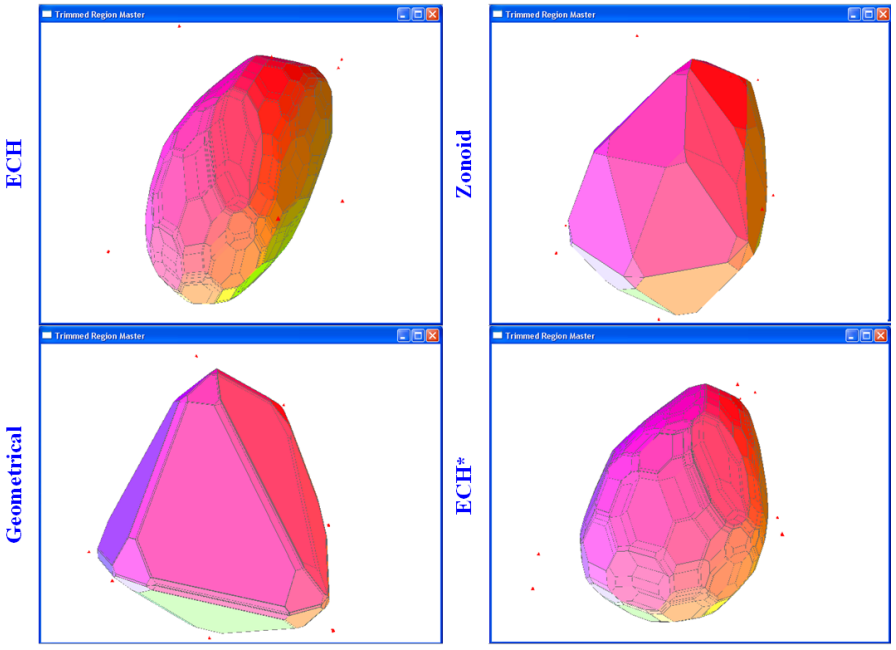

FIGURE 3.8: 3*d*-visualization of various types of WM regions.

```
( (1.290;9.249;2.059;) (0.995;9.108;1.729;) (1.099;9.129;1.662;)
(1.978;9.391;1.613;) (2.030;9.416;1.671;) (1.445;9.296;2.050;)  ).
```

For some applications it makes sense to consider only the lower or upper boundary of the WM region. This information is contained in two auxiliary files:

- `"TRegion_bound.dat"`. The same as `"TRegion.dat"` but containing facets only from the lower or upper boundary of the WM region.
- `"TRegion_vertices_bound.dat"`. The same as `"TRegion_vertices.dat"` but containing facets only from the lower or upper boundary of the WM region.

3.6 Examples

As an illustration how the algorithm works we present a comparative example of four different types of weighted-mean trimmed regions for the same data and depth parameter ($\alpha = 0.221$). Their visualization was done by a separately compiled *C++* program and is exhibited in Figure 3.8.

In this subsection we give two examples of how to get such results by means of the installed package WMTregions.

3.6.1 Illustration with simulated data

As a first example, we show how to use the package on a randomly generated sample input file. Suppose we want to calculate a zonoid region of `100` data points in $\mathbb{R}^3$ with depth `0.117`. First, we load the package:

```
R> library("WMTregions")
```

```
Loading required package: rggobi
Loading required package: RGtk2
Loading required package: rgl
```

Then we provide an input file with the data. The simplest way here is to use an embedded function `generTRsample()`. Having generated the file, we start the main procedure of calculating the WM region:

```
R> generTRsample("Cloud.dat", dim = 3, num = 100, alpha = 0.117, trtype = "zonoid")
R> WMTR("Cloud.dat")
```

```
[1] "The trimmed region was successfully calculated!"
```

Now we have two possibilities of visualizing the results: `showWMTR()` or `visualWMTR()`. As the data has dimension $d = 3$, the most appropriate choice is `visualWMTR()`:

```
R> visualWMTR()
```

You can see the $3d$-picture on the left side of Figure 3.9. On a color screen, the demonstrated facets are blue, while the ridges of the trimmed region are drawn in light green. Small red spheres represent the data cloud points. You can rotate or zoom the picture easily with the mouse.

Having obtained the result, we might want to check whether some point, say the origin $\mathbf{0}'$, lies inside the WM region. The corresponding check is conducted as follows:

```
R> tregion <- loadWMTR( "TRegion.dat" )
R> point2check = c(0,0,0)
R> pointinTR( point2check, tregion )
```

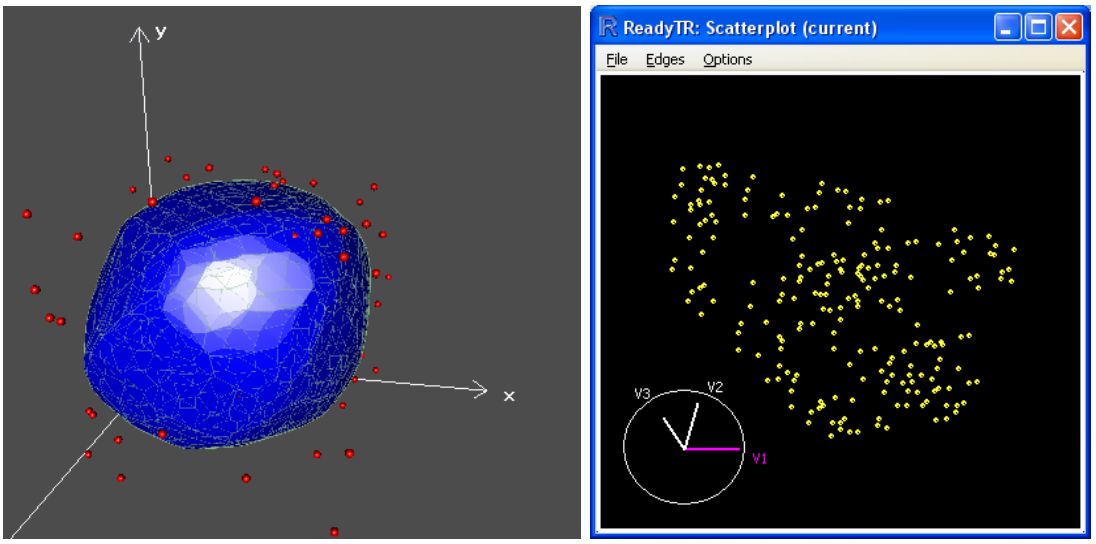

FIGURE 3.9: Visualization of the results in *R*.

```
[1] FALSE
```

Thus, the origin is outside the calculated WM region. The right side of Figure 3.9 gives rggobi based visualization of the results for the data in dimension 4:

```
R> genertrsample("Cloud4.dat", dim = 4, num = 25)
R> WMTR("Cloud4.dat")

[1] "The trimmed region was successfully calculated!"

R> showWMTR()
```

Concerning available tools for manipulating the visualization in the window, we refer the reader to rggobi documentation.

3.6.2 Calculating multivariate set-valued risk measures

The second example represents an application of the WM regions to the risk management. Our aim is to calculate the multivariate expected shortfall (Cascos and Molchanov, 2007) set-valued risk measure. We have chosen this measure because it is the most important coherent risk measure. A zonoid region $ZD_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ determines the expected shortfall at the level α as $ES_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \mathbb{R}^d \setminus (ZD_\alpha(\mathbf{x}_1, \ldots, \mathbf{x}_n) \oplus \mathbb{R}^d_+)$. In other words, it is determined by the lower boundary of the zonoid region.

```
R> library("WMTregions")
```

```
Loading required package: rggobi
Loading required package: RGtk2
Loading required package: rgl
```

We have a file `"Indices_0809.dat"` with the real life data representing the relative losses in percent on DAX (x variable), Dow Jones (y variable) and Hang Seng (z variable) stock market indices in the years 2008 and 2009. The file can be downloaded from the Web page `http://www.uni-koeln.de/wiso-fak/wisostatsem/Forschung/WMT_Regions`. Besides this, the data cloud points coordinates are also stored in the data set `"Indices_0809"` attached to the package. Using it as a historical information, we have to calculate the expected shortfall at the level 1% of the portfolio on these three indices. The data is represented by losses, therefore we are seeking for the reverse, that is, the upper boundary:

```
R> WMTR("Indices_0809.dat", bound = 1)

[1] "The trimmed region was successfully calculated!"

R> visualWMTR()
```

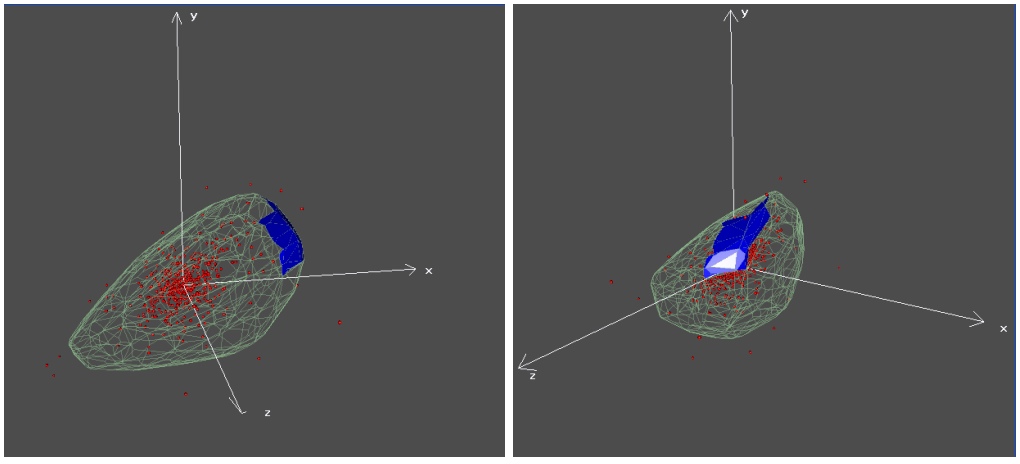

FIGURE 3.10: Visualization of the results in *R*.

On Figure 3.10 we see the most critical part of the surface in blue. It is shown from two sides.

3.7 Conclusions

An exact algorithm has been constructed to compute the WM regions of an empirical distribution in d-space for an arbitrarily given weight vector. It calculates all facets,

edges, and vertices of a region at any given depth $\alpha \in]0,1[$. (Recall that $\alpha = 0$ and $\alpha = 1$ are trivial cases.) In fact, the case $\alpha = 0$ can be also calculated, but then the WM region degenerates into the convex hull of the data set.

The *"double-hash"* mechanism plays the prominent role by marking in a special way the ridges in the hash table, thus guaranteeing that each facet of the WM region is generated only once and only these facets are calculated. It induces the unique order on the set of the facets, making the algorithm efficient. Really, the latter has as many loops as the WM region has facets, which, obviously, is the minimum number of facet generating loops in this sort of algorithms. Moreover, in some significant applications of the WM regions, such as the multivariate risk measurement, we can take advantage of the connectivity of the generated facets and calculate only, for instance, the lower boundary of the WM region, which covers about $\frac{1}{2^d}$ of its surface.

To sum up, we would like to point out some perspectives of the future work on the algorithm and the *R* package. While the precise algorithm is efficient and has the optimal number of computational steps, its most important use is to employ it as a benchmark for computationally cheaper approximate procedures. As we have seen above, WM regions have very large numbers of facets. Next possible steps of research should target at developing procedures of filtering them and replacing the "jumps" by "long jumps" across parallel ridges.

3.8 Heuristics for determining all adjacent facets

Given a basis $\mathcal{V}_F$ of a facet F, let $\mathbf{A}$ be a nonsingular $d \times d$ matrix that contains the basis vectors as its first $d-1$ rows and an arbitrary last row that is linearly independent from the other rows. Consider the linear equation

$$\mathbf{A}\mathbf{r} = \mathbf{e}_d := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \tag{3.11}$$

The unique solution $\mathbf{r}$ to this equation is a scalar multiple of the normal vector $\mathbf{p}_F$ of F.

In search for a neighboring facet, the support vector has to be rotated in a plane of dimension two in $\mathbb{R}^d$ (step C.(b.)II.v.). To reduce the algorithmic complexity of this step we compute the rotation plane in the following efficient way.

The transition from F to a neighboring facet, say via a ridge m, is done by a basis exchange. This means replacing some k rows of the matrix $\mathbf{A}$ (having indices $i \in \mathcal{I}$) by $k-1$ other data vectors and, as its last row, some vector that is linear independent from all previous rows and non-orthogonal to $\mathbf{p}$, for example $\mathbf{p}$ itself. Let $\mathbf{S}_m$ denote the $d \times d$ matrix obtained from $\mathbf{A}$ by this exchange, and $\mathbf{V}_m$ the $d \times k$ matrix built from the k new vectors as columns. Thus, the solution $\mathbf{z} = \mathbf{z}_m$ of the linear equation

$$\mathbf{S}_m \mathbf{z} = \mathbf{e}_d \,, \tag{3.12}$$

spans, together with $\mathbf{p}_F$, a plane in which the support vector $\mathbf{p}$ may be rotated.

Note that (3.12) can be solved directly by calculating $\mathbf{S}_m^{-1}$, which is the straightforward computation and has complexity $\mathcal{O}(d^3)$. Instead, in our algorithm we decompose $\mathbf{S}_m$ in order to reduce the complexity of this step. It is easy to see that

$$\mathbf{S}_m = \mathbf{K}_m \cdot \mathbf{A} \,,$$

where $\mathbf{K}_m$ is an identity matrix with substituted rows of indices $i \in \mathcal{I}$. Let these rows form a matrix $\mathbf{C}_m$ whose i-th row corresponds to the j_i-th row of $\mathbf{K}_m$. Then it holds $\mathbf{A}'\mathbf{C}_m = \mathbf{V}_m$ and, consequently,

$$\mathbf{C}_m = \left(\mathbf{A}^{-1}\right)' \mathbf{V}_m \,.$$

Note that $\mathbf{A}^{-1}$ has to be computed only once at each facet. Given $\mathbf{A}^{-1}$, calculating $\mathbf{C}_m$ has complexity $\mathcal{O}(d^2)$.

Henceforth we denote the elements of $\mathbf{A}$ and $\mathbf{C}$ by a_{ij} and c_{ij} respectively. Consider rewriting (3.12) as

$$\mathbf{K}_m \mathbf{A} \mathbf{z} = \mathbf{e}_d \,,$$

where $\mathbf{K}_m = \begin{pmatrix} 1 & 0 & & 0 \\ 0 & 1 & \ddots & 0 \\ \cdots & & & \\ c_{j_i 1} & c_{j_i 2} & c_{j_i i} & c_{j_i d} \\ \cdots & & \ddots & \\ 0 & 0 & & 1 \end{pmatrix}$ and $\mathbf{K}_m^{-1}\mathbf{e}_d = \begin{pmatrix} 0 \\ \vdots \\ -\frac{c_{j_i d}}{c_{j_i i}} \\ \vdots \\ 0 \\ 1 \end{pmatrix}$. Then it

holds

$$\mathbf{z}_m = \mathbf{A}^{-1}\mathbf{K}_m^{-1}\mathbf{e}_d = \mathbf{A}^{-1}\begin{pmatrix} 0 \\ \vdots \\ -\frac{c_{j_i d}}{c_{j_i i}} \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

For a single facet we have to compute (in $\mathcal{O}(d^3)$)

$$\mathbf{A}^{-1} = \begin{pmatrix} \boldsymbol{\alpha}_1 & | & \dots & | & \boldsymbol{\alpha}_i & | & \dots & | & \boldsymbol{\alpha}_d \end{pmatrix}.$$

Thus, besides computing $\mathbf{C}_m$, only the following computation is done to find a basis for the m-th "jump":

$$\mathbf{z}_m = \boldsymbol{\alpha}_d - \sum_{i \in \mathcal{I}} \frac{c_{j_i d}}{c_{j_i i}} \boldsymbol{\alpha}_i, \quad \mathcal{O}(d). \tag{3.13}$$

Recall that a basis for an m-th jump is given by $\{\bar{\mathbf{z}}_m, \bar{\mathbf{p}} = \bar{\mathbf{r}}\}$. Let us denote the number of ridges for a facet F by $R(F)$. The complexity of finding bases for all jumps from the facet is

$$\mathcal{O}(d^3 + (d^2 + d) \cdot R(F)) \cong \mathcal{O}(d^2 \cdot R(F)).$$

It can be easily checked that, if we do not exploit the common information on $\mathbf{A}^{-1}$, the complexity amounts to $\mathcal{O}(d^3 \cdot R(F))$.

Chapter 4

Multivariate Best-Decision Risk Measures: An Application to Portfolio Optimization

In this chapter, we consider a vector-valued multivariate risk measure that depends on the user's profile given by the user's utility. It is constructed on the basis of weighted-mean trimmed regions and represents the solution of an optimization problem. The key feature of this measure is convexity. We apply the measure to the portfolio selection problem, employing different measures of performance as objective functions.

4.1 Motivation

Quantifying risk is one of the most important problems in modern economics. Classical tools of mathematical finance include *risk measures*. These functions, as their name states, assess the risk of some financial positions, which are traditionally modeled by some random vector X. The basic idea of a risk measure is to indicate a critical value of a (monetary) deposit, or reserve, that, being added to an uncertain position, does cancel its risk in some sense. The latter means that the location of the distribution of the corresponding random vector satisfies certain formal requirements that are provided by, say, a regulator.

For example, if X is univariate, he or she may require that some α-quantile of the distribution be non-negative. If we add a constant $-Q_X(\alpha)$, which can be interpreted as an insurance deposit, to the distribution, where Q_X denotes the quantile function of X, we make the condition hold. In other words, only the worst $\alpha \cdot 100\%$ of outcomes

of the insured position are expected to be negative. In such a manner we get a famous and widely used (cf. Jorion, 2006) risk measure called *value-at-risk* (V@R). Actually, there is a plentiful assortment of different notions of risk measures, each controlling particular aspects of the outcome distribution. There is also a list of desired properties of such functions: we will refer to some of them below. As further examples of univariate risk measures, one can recall the *expected shortfall*[1], *expected minimum*, *entropic risk measure* and others.

A univariate risk measure concerns an investment into one asset. However, in practice a user is usually operating with several different assets. In this case, measuring the risk becomes a much more complicated problem than just undertaking measurements for each asset individually. The issue lies in a dependence between the assets, which can be rather complex and lead to an asymmetric joint distribution of the assets' returns.

The higher dimension is, namely the number of assets, the more importance has to be given to the dependency information. This is similar to modeling returns by a d-dimensional random vector X instead of d separate univariate random variables.

At this point, we immediately get an issue: again, the risk of X could not be comprehensively described only by risks of its marginals. To tackle this problem, a rather natural idea has been proposed. If a univariate monetary risk measure describes the minimal deterministic amount of money that, being added to the investment, compensates its risk, one could do the same in the multivariate case. In other words, we seek to find such 'minimal' deterministic vectors in $\mathbb{R}^d$ that compensate the risk of X. To get rid of the ambiguous 'minimal' qualifier, we can just take *all* deterministic vectors compensating the risk. It is easy to see that these vectors form a set in $\mathbb{R}^d$, of an affine dimension d in general. Obviously, 'minimal' vectors are lying on the surface of this d-dimensional body. In general, if there are transaction costs, there are many incomparable 'minimal' vectors.

The above gives way to *set-valued* risk measures, which are nowadays rather common in considering multivariate risks.

Working with such measures is a rather complex task, however, it becomes simpler if all such sets, that is to say, values of a set-valued risk measure, are convex. Further on, we consider this property as an advantage.

The development of coherent risk measures (Artzner et al., 1999, Delbaen, 2002) and of the pertaining machinery (see e.g. Föllmer and Schied (2004)) as well as tightening of economic standards have led to considering multivariate risks and extending the notion of

[1]It is also called the *average value-at-risk* or the *tail value-at-risk* in the literature.

the risk measure as a real-valued function to a class of set-valued functions (Jouini et al., 2004). Recently, the corresponding theory has been deeply developed both generally (see e.g. Hamel and Heyde (2010), Hamel et al. (2011), Rüschendorf (2013)) and concerning specific exemplars of risk measures (e.g. Cousin and Di Bernardino (2013), Hamel et al. (2013)).

Such literature proposes several ways of defining set-valued risk measures. In this chapter, we pursue the approach of Cascos and Molchanov (2007), who explore a direct connection of such measures to data central regions. This gives us an advantage of applying geometrical algorithms from previous chapters to calculating set-valued risk measures. In fact, computability of set-valued risk measures is usually a hard issue (cf. Hamel et al. (2014, 2013)).

The investigation of multivariate risk measures develops in several *major* tasks. The first one, the representation, is connected with a discussion of a set of *desirable properties* for a risk measure, which the reader can find, e.g., in Rachev et al. (2008). A specialized analysis of *comonotonic* risk measures is given by Ekeland et al. (2012). A widely-used *dual representation* of risk measures via acceptance sets is comprehensively described, for example, in Hamel and Heyde (2010). The second task, again, computability, is a very recent one and concerns mostly applying methods from vector optimization, such as Benson's algorithm (cf. Schrage and Löhne (2013), Hamel et al. (2014)). In this research, the key point is the computability via efficient *geometrical representations*. Besides this, the measures are intended to be applied not only in the sphere of finance but also in completely different spheres, which will be the main topic of the further chapters of this thesis.

4.2 Vector-valued multivariate risk measure based on data trimmed regions

In this section, we define a measure combining the objective evaluation of the risk by means of a set-valued risk measure, and the subjective preferences of the user, which are modeled by the user's admissable set.

4.2.1 The measure

According to Cascos (2009), using the ideas from Cascos and Molchanov (2007), a risk measure μ^d based on some data trimmed region D^*_α can be defined as follows:

$$\mu^d(X) = -\Big(D^*_\alpha(X) \oplus \mathbb{R}^d_+\Big),$$

meaning a reflection of the set $D^*_\alpha(X) \oplus \mathbb{R}^d_+$. In simple words, it states that for all $\mathbf{z} \in \mu^d$ the trimmed region $D^*_\alpha(X + \mathbf{z})$ does not lie in the positive orthant. For example, if D^*_α is a *halfspace region*, we obtain a *multivariate quantile*, which enables us to get a set-valued generalization of the *value-at-risk*. The *subadditivity* property and the analytical simplicity of *zonoid regions* enable us to use them for generalization of the *expected shortfall*, which is a *coherent* risk measure. In turn, the *expected minimum*, also a coherent measure, is generalized by means of *expected convex hull regions*.

In the same manner, we define a special class of multivariate risk measures based on weighted-mean trimmed regions $D_{\mathbf{w}_\alpha}$ given by a weight vector $\mathbf{w}_\alpha$.

Definition 4.1. The multivariate set-valued distortion risk measure is defined as follows:

$$\mu^d(X) = -\Big(D_{\mathbf{w}_\alpha}(X) \oplus \mathbb{R}^d_+\Big) \subset \mathbb{R}^d. \tag{4.1}$$

In the next chapter, we will show, why the measure is called a distortion risk measure and how it is connected to its univariate counterpart. In the current chapter, we are only interested in a measure with desirable properties, such as the subadditivity, which encourages diversification and is crucial in risk management.

We should mention that this is not a unique way of defining a multivariate distortion risk measure. For comparison, Rüschendorf (2013) gives a different notion of such a measure, which is scalar-valued: For a d-variate distribution having p.d.f. F, he considers the level set $Q(t)$ of F at level t and defines some scalar measure of $Q(t)$ as the t-quantile. Then, based on these scalar-valued quantiles, he introduces multivariate risk measures in the same way as univariate ones. Obviously, much information is lost in this case and the choice of the scalar measure is not straightforward.

To flexibilize our definition by incorporating the information about the user's preferences, described by his or her strictly increasing utility function $U(\cdot)$, we introduce the *admissable set* $\mathcal{F}$. This set collects all such returns that are perceived positively by the user. To relate it to the utility function, we assume $\mathcal{F} = \{\mathbf{y} \in \mathbb{R}^d : U(\mathbf{y}) \geq u_0\}$. Thus, the surface of $\mathcal{F}$ is the u_0-level set of the utility function.

We take the natural assumption of the user's risk aversion, which is equivalent to possessing a convex admissable set $\mathcal{F}$ (see, e.g., Föllmer and Schied (2004)). As an approximation, we suppose $\mathcal{F}$ to have the following form:

$$\mathcal{F} = \{\mathbf{y} \in \mathbb{R}^d : \mathbf{p}_k'\mathbf{y} \geq \delta_k, \quad k = 1 \dots K\} \tag{4.2}$$

with some $\mathbf{p}_1, \dots, \mathbf{p}_K \in \mathbb{R}^d_+$ and $\delta_1, \dots, \delta_K \in \mathbb{R}$, that is, $\mathcal{F}$ is an *upper convex polytope*.

E.g., a market with proportional transaction costs $\mathcal{F}$ is a cone with the apex at $\mathbf{0}$. Each level set of $U(\cdot)$ is the same (but translated) cone.

Our idea lies in a comparison of the position of the set-valued measure μ^d with that of the admissable set $\mathcal{F}$.

Definition 4.2. $\nu(X)$, a real-valued risk measure of a risky position X given the user's utility $U(\cdot)$, is defined as follows:

$$\nu(X) = \operatorname*{arg\,min}_{\mathbf{z} \in \mathbb{R}^d} \|z\|_U : \{-\mu^d(X) + \mathbf{z}\} \subset \mathcal{F}, \tag{4.3}$$

where $\|\cdot\|_U$ denotes a proper norm.

In other words, $\nu(X)$ is (in the sense of the norm $\|\cdot\|_U$) the shortest vector $\mathbf{z}$ that brings the set-valued measure $\rho(X)$ into the admissable set $\mathcal{F}$. The conventional Euclidean norm $\|\cdot\|_2$ is a natural choice for $\|\cdot\|_U$, however, a weighting of dimensions is possible due to their different importance in the user's subjective perception. If this mutual weighting is described by some nonnegative matrix Γ_U, then for any $\mathbf{z} \in \mathbb{R}^d$ it holds $\|\mathbf{z}\|_U = \|\Gamma_U \mathbf{z}\|_2$.

$\nu(\cdot)$ enjoys a clear interpretation as a *monetary* measure: The minimal reserve to be added to the position to make it acceptable. While the measure is determined by the optimal decision for the user, we will call $\nu(\cdot)$ a *best-decision risk measure*.

The transition from the set-valued measure $\mu^d(\cdot)$ to the vector-valued $\nu(\cdot)$ is realized by solving an *optimization problem*. In fact, what we are doing is a specific scalarization of a set-valued risk measure (cf. Hamel and Heyde (2010), or Schrage (2015)). Our approach consists in the most broad employment of the user profile information (given by the utility function $U(\cdot)$) for the scalarization.

It is easy to see that (4.3) in Definition 4.2 is equivalent to the following:

$$\nu(X) = \operatorname*{arg\,min}_{\mathbf{z} \in \mathbb{R}^d} \|z\|_U : \{D_{\mathbf{w}_\alpha}(X) + \mathbf{z}\} \subset \mathcal{F}. \tag{4.4}$$

Finally, we like to mention that using the measure $\nu(\cdot)$, we can define an order on a set of appropriate risky positions $\mathcal{X}$.

Definition 4.3 (Ordering risks)**.** The preference relation $\succcurlyeq_\nu$ on $\mathcal{X}$ is given as follows:

$$\forall Y, Z \in \mathcal{X} \qquad Y \succcurlyeq_\nu Z \qquad \Longleftrightarrow \qquad \|\nu(Z)\|_U \geq \|\nu(Y)\|_U.$$

4.3 Portfolio choice as a special case

A portfolio choice problem can be stated using risk measures. Unlike standard portfolio theory, where variances are used as proxies for risk, the risk measures machinery allows to treat risk more comprehensively. It is worth to mention that the disadvantage of representing risk by the variance has become a vital issue in the literature of the last decade. Besides this, two-stage mean-variance procedures, where on the first stage parameters of a model should be estimated, such as covariance matrix of random returns, are subject to estimation risk (cf. Meucci, 2009). Authors from various mathematical fields propose approaches for solving the problem. For instance, Fabozzi et al. (2010) give a detailed review of robust methods emerged in portfolio optimization and the corresponding literature. These methods are usually based on modeling uncertainty either in parameters (e.g., Costa and Paiva (2002), El Ghaoui et al. (2003), Tütüncü and Koenig (2004)) or in the whole distribution (e.g., Calafiore, 2007) and appropriate modifications of the variance. A part of recent approaches consider risk measures (e.g., Bion-Nadal and Kervarec (2012), Drapeau and Kupper (2013), Rockafellar et al. (2006)). A qualitatively new algorithm, which efficiently combines robust optimization with coherent risk measures, contributing to this trend, has been proposed by Mosler and Bazovkin (2014). This approach will be considered in Section 5.1 of the next chapter. In this chapter, we solve the portfolio choice problem using optimization of either the multivariate risk measure $\nu(\cdot)$ or some *performance measure*. In our approach, we get rid of usual distributional assumptions on returns, namely their *ellipticity*.

Let $\tilde{r}_1, \ldots, \tilde{r}_d$ be random return rates on d assets. We will notate $\tilde{\mathbf{r}} = (\tilde{r}_1, \ldots, \tilde{r}_d)'$. A convex combination of the assets' returns is sought, $\tilde{\mathbf{r}}'\boldsymbol{\omega} = \sum_{j=1}^{d} \tilde{r}_j \omega_j$, that maximizes some performance measure. Let us have a portfolio of d assets and the historical information about its returns $\{\mathbf{r}^1, \ldots, \mathbf{r}^n\} \subset \mathbb{R}^d$. Now we can consider a task of finding a portfolio with the lowest risk possible or a portfolio optimized by means of some generalized performance measure, for example, a Sharpe ratio.

To solve the task, we use the multivariate measure $\nu(\cdot)$ given by Definition 4.2 in the previous section. The considered problem has a certain form of the admissable set: a halfspace, that is, a special case of (4.2). The border of the halfspace, a hyperplane,

is determined by a portfolio vector, or simply a *portfolio*, $\boldsymbol{\omega} \in \Delta^d = \{\boldsymbol{\delta} \in \mathbb{R}^d : \boldsymbol{\delta} \geq \mathbf{0}, \mathbf{1}'\boldsymbol{\delta} = 1\}$, with no short selling permitted. This fact enables us to control the admissable set by means of varying $\boldsymbol{\omega}$ and find one that produces the minimal risk in such a way. Thus we obtain a parametric optimization problem in the sense of optimizing the risk measure $\nu(\cdot)$ or some function dependent on it. In the following subsection we propose an efficient geometric procedure of finding the optimal $\boldsymbol{\omega}^{\text{opt}}$ in the space $\mathbb{R}^d$.

4.3.1 Minimal risk portfolio

Let $d \times d$ matrix $\boldsymbol{\Omega} = \text{diag}(\boldsymbol{\omega})$. We are minimizing the risk of a portfolio, that is, are employing the following criterion $g(\boldsymbol{\omega})$:

$$g(\boldsymbol{\omega}) = \|\nu(\boldsymbol{\Omega}\tilde{\mathbf{r}})\|_U \to \min_{\boldsymbol{\omega} \in \Delta^d} . \tag{4.5}$$

Note that later the restriction to $\boldsymbol{\omega} \in \Delta^d$ will be relaxed concerning the non-negativity of components.

For a specified distortion risk measure, namely a given weight vector $\mathbf{w}_\alpha$, and an empirical sample $\mathbf{r}^1, \ldots, \mathbf{r}^n$, we construct a trimmed region $D_{\mathbf{w}_\alpha}(\mathbf{r}^1, \ldots, \mathbf{r}^n)$.

Taking the Euclidean norm as $\|\cdot\|_U$, the value of the objective $g(\boldsymbol{\omega})$ for some $\boldsymbol{\omega}$ is the Euclidean length of the minimal shift $\mathbf{s}_{\boldsymbol{\omega}} \in \mathbb{R}^d$ of the admissable set $\mathcal{F} = \{\mathbf{x} : \mathbf{1}'\mathbf{x} \geq 0\}$ such that for the data weighted by $\boldsymbol{\omega}$ it holds:

$$D_{\mathbf{w}_\alpha}(\boldsymbol{\Omega}\mathbf{r}^1, \ldots, \boldsymbol{\Omega}\mathbf{r}^n) \subset \mathcal{F} - \mathbf{s}_{\boldsymbol{\omega}}. \tag{4.6}$$

Obviously, $\mathbf{s}_{\boldsymbol{\omega}} = \nu(\boldsymbol{\Omega}\tilde{\mathbf{r}})$, where $\tilde{\mathbf{r}}$ is empirically distributed on $\mathbf{r}^1, \ldots, \mathbf{r}^n$. For convenience, we will denote $\mathcal{F} - \mathbf{s}_{\boldsymbol{\omega}}$ by $\widehat{\mathcal{F}}_{\boldsymbol{\omega}}$.

Let us now do an inverse transform of the space, that is, a linear transform by $\boldsymbol{\Omega}^{-1}$. Then, we get $\boldsymbol{\Omega}^{-1}\mathcal{F}$ instead of $\mathcal{F}$ and, respectively, the condition (4.6) becomes equivalent to the following:

$$D_{\mathbf{w}_\alpha}(\mathbf{r}^1, \ldots, \mathbf{r}^n) \subset \boldsymbol{\Omega}^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}}. \tag{4.7}$$

Because of the budget constraint $\mathbf{1}'\boldsymbol{\omega} = 1$, it is easy to show that the harmonic mean of axes intersections with the hyperplane $\partial\widehat{\mathcal{F}}_{\boldsymbol{\omega}}$ does not change after getting to $\partial\{\boldsymbol{\Omega}^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}}\}$. It equals $g(\boldsymbol{\omega})\sqrt{d}$, where $\partial\{\cdot\}$ denotes the border of a set. Now, let some $\boldsymbol{\omega}^1, \boldsymbol{\omega}^2$ produce the same objective values $g(\boldsymbol{\omega}^1) = g(\boldsymbol{\omega}^2) = g$ and form the borders $\partial\{\boldsymbol{\Omega}_1^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}^1}\}$ and $\partial\{\boldsymbol{\Omega}_2^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}^2}\}$, respectively. It can be shown that these borders intersect at the point $-\mathbf{s}_{\boldsymbol{\omega}^1} = -\mathbf{s}_{\boldsymbol{\omega}^2} = (-\frac{g}{\sqrt{d}}, \ldots, -\frac{g}{\sqrt{d}})'$. This point delimits the interval $(\mathbf{0}; -s_{\boldsymbol{\omega}^1})$ on the bisector, which has length g.

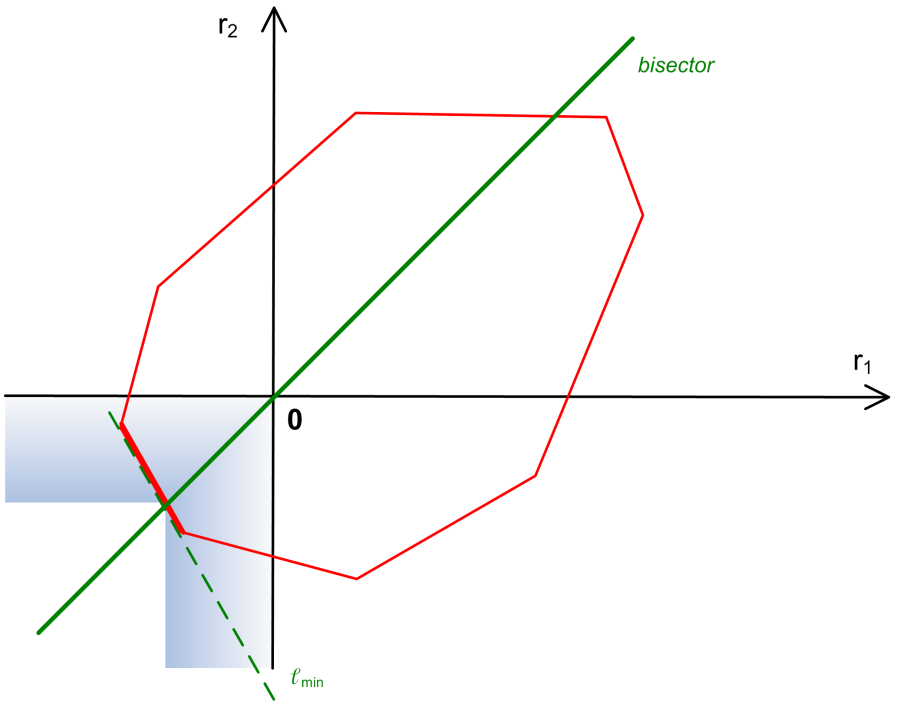

FIGURE 4.1: Searching the minimal risk portfolio.

Thus, we see that there is a bijection between all plausible $\boldsymbol{\omega}$-s and $\boldsymbol{\Omega}^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}}$. Moreover, $g(\boldsymbol{\omega})$ is the length of the interval cut off by the surface $\partial\{\boldsymbol{\Omega}^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}}\}$ on the bisector. Hence we have to find a hyperplane $\partial\{\boldsymbol{\Omega}^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}^{\text{opt}}}\}$ that 'covers' $D_{\mathbf{w}_\alpha}$ and cuts the shortest interval on the bisector. It is easy to show that it is a hyperplane $\ell_{\min}$ containing the facet intersected by the bisector (see Figure 4.1). Hence the solution is the following: the sought-for $\boldsymbol{\omega}^{\text{opt}}$ is the normalized (to the component sum of 1) normal to the facet intersected by the bisector.

4.3.2 Portfolio selection with a generalized Sharpe ratio

Now we solve the problem of maximizing the ratio of expected returns to the risk taken. It stands on the same principle as the well-known Sharpe ratio (Sharpe, 1966) and will be denoted by $\mathsf{SR}_{\tilde{\mathbf{r}}}$:

$$\mathsf{SR}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}) = \frac{\boldsymbol{\omega}' \cdot \boldsymbol{\mu}(\tilde{\mathbf{r}})}{\|\nu(\boldsymbol{\Omega}\tilde{\mathbf{r}})\|_U}\,, \tag{4.8}$$

where $\boldsymbol{\mu}(\tilde{\mathbf{r}})$, or simply $\boldsymbol{\mu}$, is the expected return $E(\tilde{\mathbf{r}})$ of the investment.

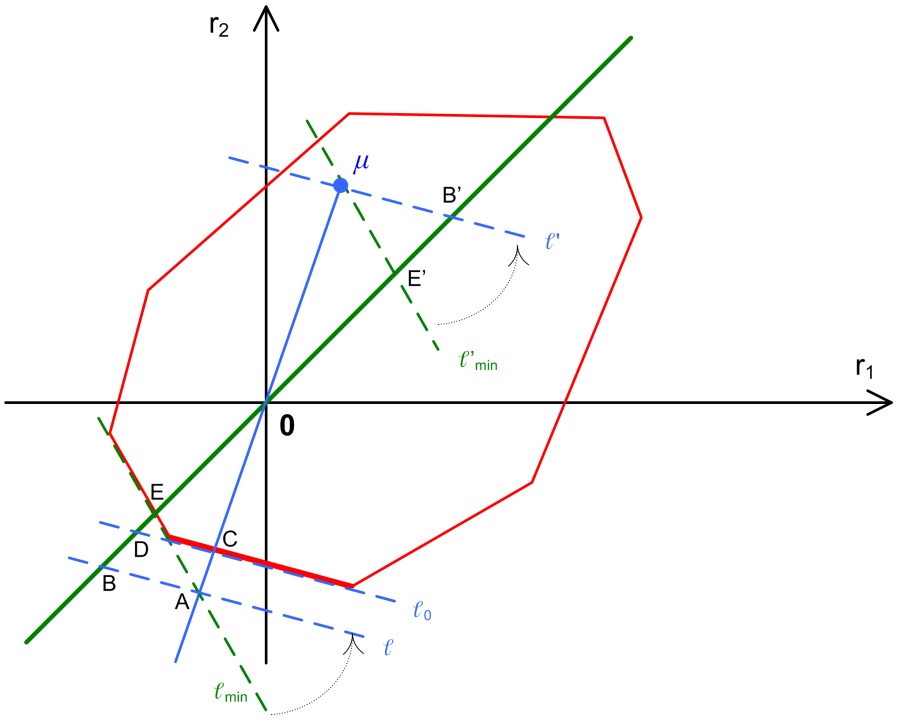

FIGURE 4.2: Searching the Sharpe ratio optimized portfolio.

4.3.2.1 Finding the optimum

Under the standard restriction on $\boldsymbol{\omega}$, $\boldsymbol{\omega} \in \Delta^d$, we have to solve the following optimization task:

$$g(\boldsymbol{\omega}) = \mathsf{SR}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}) \to \max_{\boldsymbol{\omega} \in \Delta^d} . \tag{4.9}$$

In the inverse transformed space, a hyperplane parallel to $\partial\{\boldsymbol{\Omega}^{-1}\widehat{\mathcal{F}}_{\boldsymbol{\omega}}\}$ is a set of same-return outputs $\mathbf{x}$. The value of this return equals the length of the origin-started segment of the bisector cut off by the hyperplane, because this segment is not effected by the transformation. Hence, the expected return of a portfolio $\boldsymbol{\omega}$ is equal to the length of a segment cut off by such a hyperplane containing $\boldsymbol{\mu}$. Consider Figure 4.2: for a case of minimal risk (see Subsection 4.3.1), this segment corresponds to $\mathbf{0}\mathrm{E}'$ ($\boldsymbol{\mu} \in \ell'_{\min}$, $\ell'_{\min} \| \ell_{\min}$, where '$\|$' means that $\ell'_{\min}$ and $\ell_{\min}$ are parallel). Let us now draw a line through the points $\boldsymbol{\mu}$ and $\mathbf{0}$ and find its intersection with the hyperplane $\ell_{\min}$ (the solution hyperplane for the minimal risk problem, a green dashed line on the Figure 4.2) - the point A. $\ell_{\min}$ cuts off the segment $\mathbf{0}\mathrm{E}$ with length equal to the risk estimate. $\ell'_{\min}$ cuts off the segment $\mathbf{0}\mathrm{E}'$ with length equal to the expected return. Thus, we obtain $\mathsf{SR}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}) = \frac{\mathbf{0}\mathrm{E}'}{\mathbf{0}\mathrm{E}}$.[2]

[2]Further in this chapter the name of a segment in a formula implies the length of the segment.

Let us now rotate $\ell_{\min}$ in $\mathbb{R}^d$ arbitrarily around the point A to some position ℓ. Of course, ℓ must not intersect $D_{\mathbf{w}_\alpha}$. $\ell'_{\min}$ is rotated parallelly around the point $\boldsymbol{\mu}$ to some hyperplane $\ell' \| \ell$. The rotation corresponds to browsing through different portfolios $\boldsymbol{\omega}$.

Thus, the points E and E′ move: E $\mapsto$ B, E′ $\mapsto$ B′. We immediately get $\triangle \mathbf{0}\mathrm{EA} \sim \triangle \mathbf{0}\mathrm{E}'\boldsymbol{\mu}$ and $\triangle \mathbf{0}\mathrm{BA} \sim \triangle \mathbf{0}\mathrm{B}'\boldsymbol{\mu}$, where under '$\sim$' we understand the similarity relationship. Hence $\mathsf{SR}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}') = \frac{\mathbf{0}\mathrm{E}'}{\mathbf{0}\mathrm{E}} = \frac{\mathbf{0}\boldsymbol{\mu}}{\mathbf{0}\mathrm{A}} = \frac{\mathbf{0}\mathrm{B}'}{\mathbf{0}\mathrm{B}} = \text{const}$. For each rotation $\boldsymbol{\omega}$ there is a hyperplane $\ell_0 \| \ell$ which touches $D_{\mathbf{w}_\alpha}$ and intersects the bisector at D, that is, gives the actual estimation $\|\mathbf{0}\mathrm{D}\|_U$ of risk of the portfolio $\boldsymbol{\omega}$.

To maximize $\mathsf{SR}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}) = \mathsf{SR}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}') \cdot \frac{\mathbf{0}\mathrm{B}}{\mathbf{0}\mathrm{D}}$, we should maximize $\frac{\mathbf{0}\mathrm{B}}{\mathbf{0}\mathrm{D}}$ by a rotation. Let C be an intersection of the line $(\mathbf{0}, \boldsymbol{\mu})$ with the hyperplane ℓ_0. Then it holds $\triangle \mathbf{0}\mathrm{DC} \sim \triangle \mathbf{0}\mathrm{BA}$, leading to $\frac{\mathbf{0}\mathrm{B}}{\mathbf{0}\mathrm{D}} = \frac{\mathbf{0}\mathrm{A}}{\mathbf{0}\mathrm{C}}$. At the same time, $\mathbf{0}\mathrm{A}$ remains constant, which means that we should just minimize $\mathbf{0}\mathrm{C}$. It is easy to see that the shortest possible $\mathbf{0}\mathrm{C}$ is the interval with the point C lying on the border of $D_{\mathbf{w}_\alpha}$. In turn, it means that the sought-for optimal hyperplane ℓ_0^{opt} is that containing the facet intersected by the line $(\mathbf{0}, \boldsymbol{\mu})$.

Hence, the sought-for *solution* $\boldsymbol{\omega}^{\text{opt}}$ is a normalized (to the component sum of 1) normal to the facet of the lower boundary of $D_{\mathbf{w}_\alpha}$ intersected by the line $(\mathbf{0}, \boldsymbol{\mu})$.

It is easy to check, that the above considerations hold for all $d \geq 2$, although being illustrated in $\mathbb{R}^2$.

The reader may make the following observations, which are quite important:

1. If all assets yield similar expected returns, the procedure calculates the minimal risk portfolio (because $\boldsymbol{\mu}$ lies on the bisector), which is intuitively natural. In this case, the procedure degenerates to one from Subsection 4.3.1.
2. The procedure can be enlarged to the case of data following a general probability distribution. The solution will be the similarly normalized vector tangent to the lower surface of $D_{\mathbf{w}_\alpha}$ at the point of its intersection with the line $(\mathbf{0}, \boldsymbol{\mu})$.
3. $\boldsymbol{\mu}$ can be replaced, for example, by a median or a shrinkage location estimator (cf. Meucci (2009)).

If we have some risk-free asset with the risk-free rate r_f, we put a point $\mathbf{u}_0$ on the bisector so that the length of the interval $[\mathbf{0}, \mathbf{u}_0]$ equals r_f. Then, it is easy to show that one should apply the same procedure as above, just replacing the line $(\mathbf{0}, \boldsymbol{\mu})$ by $(\mathbf{u}_0, \boldsymbol{\mu})$, likewise changing the focus of the intersecting ray.

At this step, it is interesting to observe how the procedure works in the special case of elliptically distributed returns with some covariance matrix $\boldsymbol{\Sigma}$. While in this case WM

regions asymptotically converge (see Dyckerhoff and Mosler (2012), Mosler (2002)) to ellipsoids with the shape matrix $\mathbf{\Sigma}$ for any choice of α and type of the region, it can be easily shown that the solution will, in turn, converge to the *tangential portfolio*:

$$\boldsymbol{\omega}^{\text{opt}} = \frac{\mathbf{\Sigma}^{-1}\boldsymbol{\mu}}{\mathbf{1}'\mathbf{\Sigma}^{-1}\boldsymbol{\mu}}. \tag{4.10}$$

Really, the normal to the tangent hyperplane at the ellipsoid's point intersected by the line of direction $\boldsymbol{\mu}$ is $\mathbf{\Sigma}^{-1}\boldsymbol{\mu}$. Having normalized the vector, we get the above formula (4.10).

It is immediately seen that replacing $\boldsymbol{\mu}$ with $\mathbf{1}$ above gives the portolio $\frac{\mathbf{\Sigma}^{-1}\mathbf{1}}{\mathbf{1}'\mathbf{\Sigma}^{-1}\mathbf{1}}$, which, in turn, is the minimal variance portfolio. It means that the latter is defined in a standard way by the intersection of a line parallel to the bisector and passing through $\boldsymbol{\mu}$ with the trimmed region.

These facts demonstrate that our approach is a generalization of a typical mean-variance procedure, where standard distributional assumptions are avoided and a comprehensive non-parametric risk measure is employed.

4.3.2.2 The algorithm

Input

- $\{\mathbf{r}^1, \ldots, \mathbf{r}^n\} \subset \mathbb{R}^d$ - the given empirical data about returns.
- Risk parameter α.
- The type of distortion risk measure μ^d to be used.
- *Optionally:* The risk-free rate r_f.

Output

- The optimal portfolio $\boldsymbol{\omega}^{\text{opt}}$.
- Value of the criterion.

Steps (SR-algorithm)

SR1. Define the weight vector $\mathbf{w}_\alpha$. Construct a focus point $\mathbf{u}_0 = (\frac{r_f}{\sqrt{d}}, \ldots, \frac{r_f}{\sqrt{d}})'$ or take, by default, the origin $\mathbf{0}$. Construct a line $\varphi = (\mathbf{u}_0, \boldsymbol{\mu})$ or $\varphi = (\mathbf{0}, \boldsymbol{\mu})$ respectively.

SR2. Calculate (Bazovkin and Mosler, 2012a) a part of $D_{\mathbf{w}_\alpha}(\mathbf{r}^1, \ldots, \mathbf{r}^n)$ in the place of a probable intersection with φ (cf. the *efficient set* in Mosler and Bazovkin (2014)). The type of $D_{\mathbf{w}_\alpha}$ corresponds to the selected distortion risk measure.

SR3. Find the facet of $D_{\mathbf{w}_\alpha}$ intersected by φ. Get the normal $\vec{n}_{\text{opt}}$ to the hyperplane containing it. The sought-for $\boldsymbol{\omega}^{\text{opt}} = \frac{\vec{n}_{\text{opt}}}{\mathbf{1}'\vec{n}_{\text{opt}}}$.

A special consideration is needed for a case when there is some negative component in $\boldsymbol{\omega}^{\text{opt}}$, namely $\exists i : \omega_i^{\text{opt}} < 0$. If it occurs, one sets $\omega_i^{\text{opt}} = 0$ and solves the task without the i-th asset (namely projecting onto $\mathbb{R}^{d-1}$). However, this situation can be managed more flexibly, which is the topic of Subsection 4.3.4 below.

The intersected facet from Step SR3. of the algorithm can be realized as follows:

A. Construct the first facet of $D_{\mathbf{w}_\alpha}(\mathbf{r}^1, \ldots, \mathbf{r}^n)$ with the normal close to the direction of φ.

B. Find the neighboring facet with the best criterion (Mosler and Bazovkin, 2014) describing its distance from φ.

C. Jump to the facet found and go to step B..

It can be seen that on each step of this subalgorithm we get a better solution. Furthermore, the tactics of the *"long jump"* can be used, where a jump over some neighbors in a criterion-enhancing direction is made at one step.

The main complexity-contributing issues are the following:

1. Calculating some facets of the trimmed region $D_{\mathbf{w}_\alpha}$: much simpler than calculating the whole region (since knowing φ).
2. Finding an intersection of a line with a convex surface in $\mathbb{R}^d$.

4.3.3 Optimization with a generalized certainty equivalent

In this subsection we pursue the same optimization problem but with a performance measure given by the certainty equivalent, which is commonly used in modern portfolio theory (cf. Markowitz (1952)). Again, the difference is that we replace the variance by the risk measure $\nu(\cdot)$. Then the criterion is the following:

$$\mathsf{CE}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}) = \boldsymbol{\omega}'\boldsymbol{\mu} - \lambda \cdot \|\nu(\boldsymbol{\Omega}\tilde{\mathbf{r}})\|_U \,, \tag{4.11}$$

where λ is a given positive constant describing the risk aversion of the user.

4.3.3.1 Finding the optimum

We will maximize $\frac{1}{\lambda}\mathsf{CE}_{\tilde{\mathbf{r}}}$, namely:

$$g(\boldsymbol{\omega}) = \frac{1}{\lambda}\mathsf{CE}_{\tilde{\mathbf{r}}}(\boldsymbol{\omega}) \to \max_{\boldsymbol{\omega}\in\Delta^d} . \tag{4.12}$$

First, we create a point $\boldsymbol{\mu}_\lambda = -\frac{1}{\lambda}\boldsymbol{\mu}$. Now consider Figure 4.3. If we have a portfolio $\boldsymbol{\omega}^1$ given by the hyperplane ℓ_1, the corresponding risk $\|\nu(\boldsymbol{\Omega}X)\|_U$ equals the length of the segment $\mathrm{A}_1\mathbf{0}$. Analogously to the previous subsection, $\frac{1}{\lambda}\boldsymbol{\omega}^{1\prime}\boldsymbol{\mu}$ equals $\mathrm{B}_1\mathbf{0}$, where $\mathrm{B}_1 = \ell_1' \cap \{\text{bisector}\}$ and ℓ_1' is a hyperplane parallel to ℓ_1 and containing $\boldsymbol{\mu}_\lambda$. Now, it is directly seen that $\frac{1}{\lambda}\mathsf{CE}_{\tilde{\mathbf{r}}}$ equals $\mathrm{A}_1\mathrm{B}_1 = \mathrm{B}_1\mathbf{0} - \mathrm{A}_1\mathbf{0}$. The same principle is applied to a portfolio $\boldsymbol{\omega}^2$, yielding the criterion value $\mathrm{A}_2\mathrm{B}_2$ for the latter.

We see that $\mathrm{A}_1\mathrm{B}_1 < \mathrm{A}_2\mathrm{B}_2$, because ℓ_1 rotates to the position ℓ_2 with a smaller shoulder relatively to (i.e. distance to) the bisector as ℓ_1' to ℓ_2'. That is to say, $\mathrm{A}_i\mathrm{B}_i$ increases while rotating ℓ_i if ℓ_i' has a larger shoulder relatively to the bisector and vice versa. Thus, starting from the minimal risk position, we first increase $\mathsf{CE}_{\tilde{\mathbf{r}}}$ until ℓ gets a pivot more distant from the bisector as $\boldsymbol{\mu}_\lambda$. It can happen when leaving a position containing a facet that, in turn, contains points equidistant with $\boldsymbol{\mu}_\lambda$ from the bisector. Hence the optimal hyperplane is one containing a facet intersected by a hypercylinder with the bisector as its axis and $\boldsymbol{\mu}_\lambda$ lying on its surface.

To construct an algorithm, we first find a facet intersected by a line parallel to the bisector and containing $\boldsymbol{\mu}_\lambda$. It is the first candidate. Then we move along the ring (intersection with the hypercylinder) and check the values of $\mathsf{CE}_{\tilde{\mathbf{r}}}$ for each of the facets. A facet ℓ_{j^*} with the maximum $\mathsf{CE}_{\tilde{\mathbf{r}}}$ defines the optimal portfolio.

Finally, consider a special case when $\lambda \to \infty$. Maximizing the criterion (4.11) becomes equivalent to optimizing $\nu(\boldsymbol{\Omega}\tilde{\mathbf{r}})$. Thus, we obtain the minimal risk problem. While $\boldsymbol{\mu}_\lambda \to \mathbf{0}$, the hypercylinder degenerates into a line. Hence the sought-for facet is the facet intersected by the bisector. Obviously, we get the same solution as in Subsection 4.3.1. Another extreme case occurs when λ is small enough, so that the hypercylinder contains $D_{\mathbf{w}_\alpha}$. In this case, we can rotate ℓ until it becomes parallel to the bisector ($\mathbf{1}'\vec{n} = 0$). Clearly, that from all such hyperplanes the optimum is given by the one that is most remote from $D_{\mathbf{w}_\alpha}$. This optimum is a vector that has a single positive component for the maximal expected return and others are negative. It means purchasing only the asset j with $\mu_j = \max\{\mu_1, \ldots, \mu_d\}$, where $(\mu_1, \ldots, \mu_d)$ $equiv\boldsymbol{\mu}$.

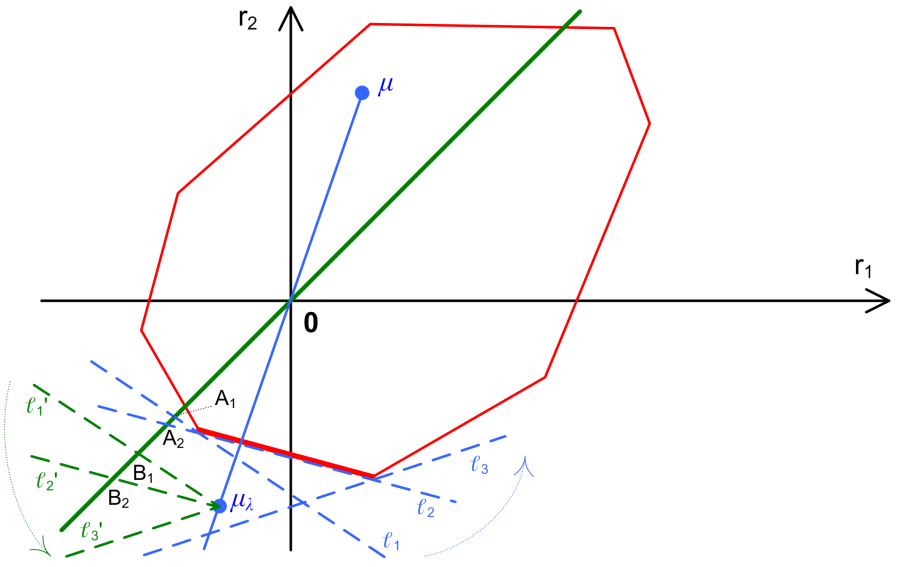

FIGURE 4.3: Searching the certainty equivalent optimized portfolio.

4.3.3.2 The algorithm

Input

- $\{\mathbf{r}^1, \ldots, \mathbf{r}^n\} \subset \mathbb{R}^d$ - the given empirical data about returns.
- Risk parameter α.
- The type of distortion risk measure μ^d to be used.
- The risk aversion constant λ.

Output

- The optimal portfolio $\boldsymbol{\omega}^{\text{opt}}$.

Steps (CE-algorithm)

CE1. Calculate the relevant part of the trimmed region $D_{\mathbf{w}_\alpha}(\mathbf{r}^1, \ldots, \mathbf{r}^n)$.

CE2. Construct the point $\boldsymbol{\mu}_\lambda = -\frac{1}{\lambda}\boldsymbol{\mu}$.

CE3. Build a line parallel to the bisector and containing $\boldsymbol{\mu}_\lambda$. Find its intersection with $D_{\mathbf{w}_\alpha}$ similarly to the step SR3. of the SR-algorithm.

CE4. Calculate the criterion $\frac{1}{\lambda}\mathsf{CE}_{\tilde{\mathbf{r}}}$ for the current facet with the index j. It equals the length of the segment $\mathrm{A}_j\mathrm{B}_j$. If it is the best currently, store the facet.

CE5. Find appropriate neighboring facets for the newly stored facet. For each of them, go to the step CE4. If there is no new neighbors, *stop*.

   a. A neighbor is appropriate if it contains points equidistant with $\boldsymbol{\mu}_\lambda$ to the bisector, which means being intersected by the hypercylinder. In doing this, calculate the min and max distances of the facet's points to the bisector.

CE6. Get a normal $\vec{n}_{\text{opt}}$ to the current best facet. The sought-for $\boldsymbol{\omega}^{\text{opt}} = \frac{\vec{n}_{\text{opt}}}{\mathbf{1}'\vec{n}_{\text{opt}}}$.

A case of negative weights can be solved as proposed in Subsection 4.3.4. If negative weights are not allowed, we pursue them analogously to Subsection 4.3.2.2.

4.3.4 Negative weights and short sellings

It is well-known that an estimated negative value of ω_i for the i-th asset actually proposes to do a short selling of that asset. We can use such a strategy as an alternative to just fixing corresponding weights to 0 and solving the similarly stated subproblem for the remaining assets. The approach given in this subsection is common for both the SR-algorithm and the CE-algorithm.

4.3.4.1 Optimum with shorting permitted

First, we modify the derivation of $\boldsymbol{\omega}^{\text{opt}}$ from a found $\vec{n}_{\text{opt}}$ due to the relaxation of the restriction $\boldsymbol{\omega} \in \Delta^d$. Namely only the sum of component absolute values $\|\boldsymbol{\omega}\|_1$ is set to 1, resulting in $\boldsymbol{\omega}^{\text{opt}} = \frac{\vec{n}_{\text{opt}}}{\|\vec{n}_{\text{opt}}\|_1}$. Let the user possess stores of the d assets available for allocating at the rates of $S_1, \ldots, S_d$ units. The idea is to solve the task recursively.

We start from all d assets and on each stage allocate a finite number of units Z_k and eliminate those assets whose store is fully exhausted on the current stage. This filtering implies setting weights to 0 for the 'bottleneck' assets on next stages. We solve the filtered task recursively until we get some stage T with an optimal solution without negative components, or there is nothing more to allocate. Now consider a stage k: let J_k be a set of indices corresponding to negative components of the optimal portfolio on this stage, $\boldsymbol{\omega}^{\text{opt}_k}$. We determine the volume of units to be allocated on this step:

$$Z_k = \min_{j \in J_k} \frac{S_j^k}{|\omega_j^{\text{opt}_k}|},$$

where S_j^k denotes an available store of the asset j at the beginning of the stage k.

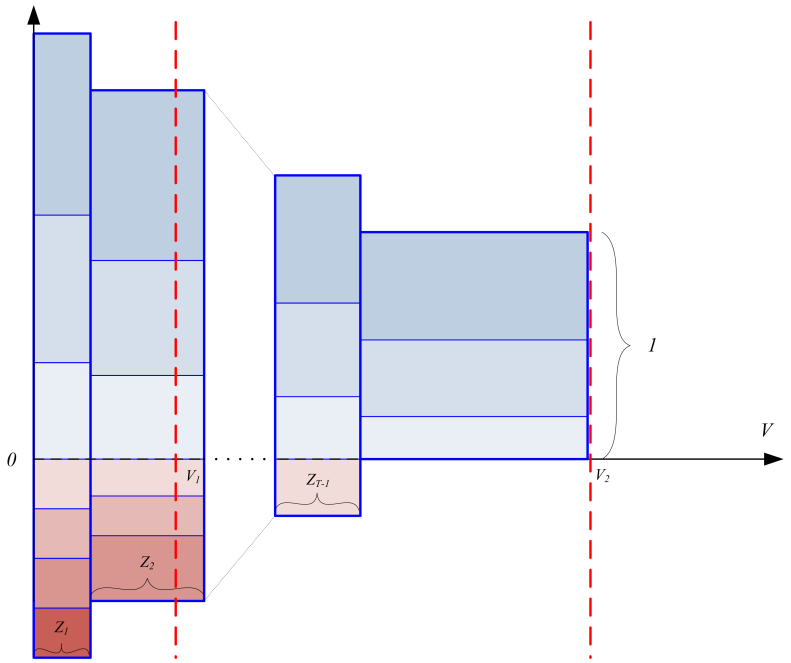

FIGURE 4.4: The recursive procedure for negative weights.

The structure of the problem is typical for *dynamic programming*, and each stage is pursued optimally. It means that the recursive procedure yields the overall optimal solution.

As a result, we obtain a 'ladder' of allocated units (see Figure 4.4) $Z_1, \ldots, Z_T$, which yields the optimal allocation after an aggregation. If we want to invest some V units, we find such K that $\sum_{i=1}^{K-1} Z_i \leq V \leq \sum_{i=1}^{K} Z_i$. Then invest[3] Z_i into $\boldsymbol{\omega}^{\mathrm{opt}_k}$ for all $k = 1, \ldots, K-1$. The rest of V we invest into $\boldsymbol{\omega}^{\mathrm{opt}_K}$. For example, on Figure 4.4, for $V = V_1$ we have $K = 2$, while for $V = V_2$, K equals T.

This simple example shows that the optimal aggregate portfolio depends on V (without shorting permitted, it is independent).

4.3.4.2 The algorithmic supplement

Input

- An aggregate number of units V to be allocated.
- Available stores $S_1, \ldots, S_d$ of the assets.
- Standard inputs for either SR- or CE-algorithm.

[3]Investing into a negatively weighted asset means shorting it.

Output

- The optimal allocation $\{V_1, \ldots, V_d\}$.

Steps (NW-supplement)

NW1. The first step (d assets, nothing invested): $k := 1$, $\hat{V} = 0, V_j = 0\ \forall j = 1 \ldots d$.

NW2. Find the optimal portfolio $\boldsymbol{\omega}^{\mathrm{opt}_k}$ by the SR- or CE-algorithm, while fixing weights of eliminated assets at zero.

NW3. $\boldsymbol{\omega}^{\mathrm{opt}_k} := \frac{\boldsymbol{\omega}^{\mathrm{opt}_k}}{\|\boldsymbol{\omega}^{\mathrm{opt}_k}\|_1}$; $J_k = \{j : \omega_j^{\mathrm{opt}_k} < 0\}$.

NW4. If $\forall j$ holds $\omega_j^{\mathrm{opt}_k} \geq 0$, then $Z_k = V - \hat{V}$; else $Z_k = \min_{j \in J_k} \frac{S_i}{|\omega_j^{\mathrm{opt}_k}|}$.

NW5. $Z_k := \min\{Z_k, V - \hat{V}\}$.

NW6. $V_j := V_j + Z_k \omega_j^{\mathrm{opt}_k}, \forall j$.

NW7. $S_j := S_j + Z_k \omega_j^{\mathrm{opt}_k}, \forall j$.

NW8. $\hat{V} := \hat{V} + Z_k$. If $\hat{V} = V$, go to Step NW10.

NW9. Eliminate assets with indices in J_k. $k := k + 1$. Go to Step NW2.

NW10. V_j is the final investment into the j-th asset. $V = \sum_j V_j$.

4.4 Discussion

In this chapter we have shown a connection between set-valued distortion risk measures and weighted-mean trimmed regions. The former can be calculated using the algorithms from previous chapters. We have considered the multivariate vector-valued risk measure $\nu(\cdot)$ that, firstly, aggregates the information from a set-valued coherent distortion risk measure and, at the same time, employs the user's risk posture information.

In a special case of substitutable components, we have applied the measure $\nu(\cdot)$ to solving a portfolio choice problem with different performance measures as objective functions. As a result, the efficient algorithms for the minimal risk, the generalized Sharpe ratio and the generalized certainty equivalent were proposed.

As a possible extension to be regarded, the shape of a trimmed region can be modified explicitly or via visual tools. The minimal risk and the SR-algorithm are realized in

an R package PortfolioTR (Bazovkin, 2013). Besides this, the framework is flexible for incorporating further possible performance measures.

One more potential way of development of the framework lies in extending it to markets with transaction costs with admissable sets in form of convex cones or convex upper polytopes (4.2). An application of the risk measure $\nu(\cdot)$ for such situations is considered in Chapter 6.

Chapter 5

Stochastic Linear Programming and Distortion Risk Measures

In this chapter, we apply coherent distortion risk measures to capture the possible violation of a restriction in linear optimization problems whose parameters are uncertain. Each risk constraint induces an uncertainty set of coefficients, which is proved to be a weighted-mean trimmed region. Thus, given a sample of the coefficients, an uncertainty set is a convex polytope that can be exactly calculated. We construct an efficient geometrical algorithm to solve stochastic linear programs that have a single distortion risk constraint. The algorithm's asymptotic behavior is also investigated, when the sample is i.i.d. from a general probability distribution. Finally, we present some computational experience.

5.1 Motivation

Uncertainty in the coefficients of a linear program is often handled by probability constraints or, more generally, bounds on a risk measure. The random restrictions are then captured by imposing risk constraints on their violation. Consider the linear program

$$\mathbf{c}'\mathbf{x} \longrightarrow \min \quad s.t.\ \tilde{\mathbf{A}}\mathbf{x} \geq \mathbf{b}\,, \tag{5.1}$$

and assume that $\tilde{\mathbf{A}}$ is a stochastic $m \times d$ matrix and $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^d$, $\mathbf{x} \in \mathbb{R}^d$. This is a stochastic linear optimization problem. To handle the stochastic restrictions a *joint risk constraint*,

$$\rho^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) \leq \mathbf{0}\,, \tag{5.2}$$

may be introduced, where ρ^m is an m-variate risk measure. For instance, with $\rho^m(Y) = \text{Prob}[Y < 0] - \alpha$ the restriction (5.2) becomes

$$\text{Prob}[\tilde{\mathbf{A}}\mathbf{x} \geq \mathbf{b}] \geq 1 - \alpha\,, \tag{5.3}$$

and a usual *chance-constrained linear program* is obtained. Alternatively, the restrictions may be subjected to *separate risk constraints*,

$$\rho^1(\tilde{\mathbf{A}}_j\mathbf{x} - b_j) \leq 0\,, \quad j = 1 \ldots m\,, \tag{5.4}$$

with $\tilde{\mathbf{A}}_j$ denoting the j-th row of $\tilde{\mathbf{A}}$. In (5.4) each restriction is subject to the same bound that limits the risk of violating the condition. A linear program that minimizes $\mathbf{c}'\mathbf{x}$ subject to one of the restrictions, (5.2) or (5.4), is called a *risk-constrained stochastic linear program.*

For stochastic linear programs (SLPs) in general and risk-constrained SLPs in particular, the reader is referred to, e.g., Kall and Mayer (2010). What we call a risk measure here is mentioned in that book as a *quality measure*, and useful representations of the corresponding constraints are given. As most of the literature, Kall and Mayer (2010) focus on classes of SLPs with chance constraints that lead to convex programming problems, since these have obvious computational advantages; see also Prékopa (1995). Our choice of the quality measure, besides its generality, enjoys a meaningful interpretation and, as it will be seen, enables the use of convex structures in the problem.

In the case of a single constraint ($m = 1$) notate

$$\rho(\tilde{\mathbf{a}}'\mathbf{x} - b) \leq 0\,. \tag{5.5}$$

A practically important example of an SLP with a single risk constraint (5.5) is the *portfolio selection problem.* Let $\tilde{r}_1, \ldots, \tilde{r}_d$ be the return rates on d assets and notate $\tilde{\mathbf{r}} = (\tilde{r}_1, \ldots, \tilde{r}_d)'$. A convex combination of the assets' returns is sought, $\tilde{\mathbf{r}}'\mathbf{x} = \sum_{j=1}^d \tilde{r}_j x_j$, that has maximum expectation under a risk constraint and an additional deterministic constraint,

$$\max_{\mathbf{x}\in\mathcal{C}} \; E[\tilde{\mathbf{r}}]'\mathbf{x}, \quad s.t.\ \rho(\tilde{\mathbf{r}}'\mathbf{x}) \leq \rho_0,\ \mathbf{x} \in \mathcal{C}\,, \tag{5.6}$$

where ρ is a risk measure, $\rho_0 \in \mathbb{R}$ is a given upper bound of risk (a nonnegative monetary value), and $\mathcal{C} \in \mathbb{R}^d$ is a deterministic set that restricts the coefficients x_k in some way. For example, if short sales are excluded, $\mathcal{C}$ is the positive orthant in $\mathbb{R}^d$. The solution $\mathbf{x}^*$ is the optimal investment under the given model. We will see that a solution, if it exists, is, as a rule, finite and unique. In our geometric approach such a solution corresponds to the intersection of some line and some convex body that both contain the point $E[\tilde{\mathbf{r}}]$.

Regarding the choice of ρ, two special cases are well known. First, let $\rho(\tilde{\mathbf{r}}'\mathbf{x}) = \text{Prob}[\tilde{\mathbf{r}}'\mathbf{x} \leq -v_0]$ and $\rho_0 = \alpha$. Then the optimization problem (5.6) says: Maximize the mean return $E[\tilde{\mathbf{r}}'\mathbf{x}]$ under the restrictions $\mathbf{x} \in \mathcal{C}$ and

$$\text{V@R}_\alpha(\tilde{\mathbf{r}}'\mathbf{x}) \leq v_0\,.$$

That is, the *value at risk* V@R_α of the portfolio return must not exceed the bound v_0. Second, let

$$\rho(\tilde{\mathbf{r}}'\mathbf{x}) = -\frac{1}{\alpha}\int_0^\alpha Q_{\tilde{\mathbf{r}}'\mathbf{x}}(t)dt\,, \tag{5.7}$$

where Q_Z signifies the quantile function of a random variable Z. This means that the *expected shortfall* of the portfolio return is employed in the risk restriction.

In practice, $\tilde{\mathbf{a}}$ has to be obtained from data. If the solution of the SLP is based on n observed coefficient vectors $\mathbf{a}^1, \ldots, \mathbf{a}^n \in \mathbb{R}^d$, the SLP is mentioned as an *empirical risk-constrained SLP*. In other words, we assume that $\tilde{\mathbf{a}}$ follows an *empirical distribution* that gives equal mass $\frac{1}{n}$ to some observed points $\mathbf{a}^1, \ldots, \mathbf{a}^n \in \mathbb{R}^d$. Rockafellar and Uryasev (2000) investigate an empirical stochastic program that arises in portfolio choice when the expected shortfall of a portfolio is minimized. They convert the objective into a function that is convex in the decision vector $\mathbf{x}$ and optimize it by standard methods. This approach is commonly used in more recent works of these and other authors on portfolio optimization.

A more complex situation is investigated by Bertsimas and Brown (2009), who discuss the risk-constrained SLP with arbitrary coherent distortion risk measures, which also include expected shortfall. These allow for a sound interpretation in terms of expected utility with distorted probabilities. For the linear restriction an, as it is called, *uncertainty set* is constructed which consists of all coefficients satisfying the risk constraint. Bertsimas and Brown (2009) discuss the uncertainty set that turns the SLP into a minimax problem, called *robust linear program*; however, they provide no optimal solution of this program there. The uncertainty set is a convex body and, as will be made precise below in this chapter, comes out to equal a so-called weighted-mean trimmed region. Natarajan et al. (2009), on the reverse, construct similar risk measures from given polyhedral and conic uncertainty sets. As an extension, Ben-Tal et al. (2010) propose the so-called "soft robustness" model, which, as they show, can be regarded as an LP with the feasible set defined by some *convex* risk measure. Such approaches are also applicable (Bertsimas and Goyal, 2013) to approximately solving a multi-stage robust convex optimization problem, where the information about the realization of uncertain parameters is adjusted on each stage.

Pflug (2006) has proposed an iterative algorithm for optimizing a portfolio using distortion functionals, on each step adding a constraint to the problem and solving it by the simplex method. Meanwhile, many other authors have recently contributed to the development of robust linear programs related to risk-constrained optimization problems: see, e.g., Nemirovski and Shapiro (2006), Ben-Tal et al. (2009) and Chen et al. (2010). For a review of robust linear programs in portfolio optimization the reader is referred to Fabozzi et al. (2010). There are also attempts to solve this problem by means of robust non-linear models (see, for instance, Kawas and Thiele (2011)), which, however, are substantially less investigated in the literature, than the linear ones. Other applications are surveyed in detail in Gabrel et al. (2014).

In this chapter we contribute to this discussion in three respects:

1. The uncertainty set of an SLP under a general coherent distortion risk constraint is shown to be a *weighted-mean trimmed region*, which provides a useful visual and computable characterization of the set.

2. An *algorithm* is constructed that solves the minimax problem over the uncertainty set, hence the SLP.

3. If the data is i.i.d. from a general probability distribution, the uncertainty set and the solution of the SLP are shown to be *consistent estimators* of the uncertainty set and the SLP solution.

The chapter is organized as follows: In Section 5.2 constraints on distortion risk measures are discussed. They are characterized by uncertainty sets in parameter, which, in turn, are shown to be weighted-mean trimmed regions (Theorem 5.2). Based on Theorem 5.2, which is a core result, we formulate a robust linear program, which is investigated in Section 5.3 and by which the SLP with a distortion risk constraint is solved. Section 5.4 introduces an algorithm for this program and discusses sensitivity issues of its solution. In Section 5.5 we address the SLP and its solution for generally distributed coefficients and investigate the limit behavior of our algorithm if based on an independent sample of coefficients. Section 5.6 contains the first computational results and concludes. The technical appendix (Section 5.7) gathers properties of distortion risk measures, a proof of Theorem 5.2 and a demonstration (Proposition 5.9) that the weighted-mean trimmed regions have the important *coherency* property.

5.2 Distortion risk constraints and weighted-mean regions

5.2.1 Distortion risk measures

A large and versatile subclass of risk measures is the class of distortion risk measures, which have appeared first from ideas in insurance research (Wang et al., 1997). Again, let Q_Y denote the quantile function of a random variable Y.

Definition 5.1 (Distortion risk measure)**.** Let r be an increasing function $[0,1] \to [0,1]$. The function ρ given by

$$\rho(Y) = -\int_0^1 Q_Y(t)dr(t) \tag{5.8}$$

is a *distortion risk measure* with *weight generating function* r.

Distortion risk measures are essentially the same as *spectral risk measures* (Acerbi, 2002)[1]. Their properties are considered in detail in Appendix 5.7. Here, we want to concentrate on their *coherency*, because of its crucial role in assessing the diversified risks.

A distortion risk measure is coherent if and only if r is concave. For example, with $r(t) = 0$ if $t < \alpha$ and $r(t) = 1$ if $t \geq \alpha$, the *value at risk* $\mathrm{V@R}_\alpha(Y) = -Q_Y(\alpha)$ is obtained, which is a non-coherent distortion risk measure. A prominent example of a coherent distortion risk measure is the *expected shortfall*, which is yielded by $r(t) = t/\alpha$ if $t < \alpha$ and $r(t) = 1$ otherwise. This measure is defined as (the negative of) the conditional expectation of Y under the condition that Y does not exceed its α-quantile, that is $Q_Y(\alpha)$, with the opposite sign. In simple words, it is the mean of the $\alpha \cdot 100\%$ biggest possible losses. Clearly, this measure is more conservative than the value at risk, because its value cannot be smaller than the corresponding value at risk. Also, given α, observe that, if $r(t) = t$, the risk measure becomes the expectation of $-Y$.

A general distortion risk measure $\rho(Y)$ can thus be interpreted as the expectation of $-Y$ with respect to a probability distribution that has been distorted by the function r. In particular, a concave function r distorts the probabilities of lower outcomes of Y in the positive direction (the lower the more) and conversely for higher outcomes (the higher the less). In empirical applications, coherent distortion risk measures other than expected shortfall have been recently used by many authors; see, e.g., Adam et al. (2008) for a comparison of various such measures in portfolio choice.

[1] Spectral risk measures coincide with coherent distortion risk measures.

An equivalent characterization of a distortion risk measure is that it is a law-invariant and comonotonic risk measure; see Kusuoka (2001). ρ is *comonotonic* if

$$\rho(Y+Z) = \rho(Y) + \rho(Z) \text{ for all } Y \text{ and } Z \text{ that are comonotonic,}$$

i.e., that satisfy $\big(Y(\omega) - Y(\omega')\big)\big(Z(\omega) - Z(\omega')\big) \geq 0$ for every $\omega, \omega' \in \Omega$. If Y has an empirical distribution on $y_1, \ldots, y_n \in \mathbb{R}$, the definition (5.8) of a distortion risk measure specializes to

$$\rho(Y) = -\sum_{i=1}^{n} q_i y_{[i]}, \tag{5.9}$$

where $y_{[i]}$ are the values ordered from above and q_i are nonnegative weights adding up to 1. (Observe that $q_i = r(y_{[\frac{n+1-i}{n}]}) - r(y_{[\frac{n-i}{n}]})$.) Then, the distortion risk measure (5.9) is coherent if and only if the weights are ordered, i.e., $\mathbf{q} \in \Delta^n_{\leq} := \{\mathbf{q} \in \Delta^n : 0 \leq q_1 \leq \cdots \leq q_n\}$.

5.2.2 Weighted-mean regions as uncertainty sets

If ρ is a coherent distortion risk measure, the uncertainty set $\mathcal{U}_\rho$ has a special geometric structure, which will be explored now in order to visualize the optimization problem and to provide the basis for an algorithm. We will demonstrate that $\mathcal{U}_\rho$ equals a so-called *weighted-mean trimmed region* (or, equivalently, *WM region*) of the distribution of $\tilde{\mathbf{a}}$.

Given the probability distribution F_Y of a random vector Y in $\mathbb{R}^d$, WM regions form a nested family of convex compact sets, $\{D_\alpha(F_Y)\}_{\alpha\in[0,1]}$, that are affine equivariant (that is $D_\alpha(F_{AY+b}) = A\, D_\alpha(F_Y) + b$ for any regular matrix A and $b \in \mathbb{R}^d$). By this, the regions describe the distribution with respect to its location, dispersion and shape. Weighted-mean trimmed regions have been introduced in Dyckerhoff and Mosler (2011) for empirical distributions, and in Dyckerhoff and Mosler (2012) for general ones.

For an empirical distribution on $\mathbf{a}^1, \ldots, \mathbf{a}^n \in \mathbb{R}^d$, a weighted-mean trimmed region is a polytope in $\mathbb{R}^d$ and defined as

$$D_{\mathbf{w}_\alpha}(\mathbf{a}^1, \ldots, \mathbf{a}^n) = \operatorname{conv}\left\{ \sum_{j=1}^{n} w_{\alpha,j} \mathbf{a}^{\pi(j)} \,:\, \pi \text{ permutation of } \{1, \ldots, n\} \right\}. \tag{5.10}$$

Here $\mathbf{w}_\alpha = [w_{\alpha,1}, \ldots, w_{\alpha,n}]'$ is a vector of ordered weights, i.e., $\mathbf{w}_\alpha \in \Delta^n_{\leq}$, indexed by $0 \leq \alpha \leq 1$ that for $\alpha < \beta$ satisfies

$$\sum_{j=1}^{k} w_{\alpha,j} \leq \sum_{j=1}^{k} w_{\beta,j}\,, \quad \forall k = 1, \ldots, n\,. \tag{5.11}$$

Any such family of *weight vectors* $\{\mathbf{w}_\alpha\}_{0\leq\alpha\leq 1}$ specifies a particular notion of weighted-mean trimmed regions. There are many types of weighted-mean trimmed regions. They contain well known trimmed regions like the zonoid regions, the expected convex hull regions and several others. For example,

$$w_{\alpha,j} = \begin{cases} \frac{1}{n\alpha} & \text{if } j > n - \lfloor n\alpha \rfloor, \\ \frac{n\alpha - \lfloor n\alpha \rfloor}{n\alpha} & \text{if } j = n - \lfloor n\alpha \rfloor, \\ 0 & \text{if } j < n - \lfloor n\alpha \rfloor, \end{cases}$$

$0 \leq \alpha \leq 1$, defines the *zonoid regions*. However, some popular types of trimmed regions, such as Mahalanobis or halfspace regions, are not weighted-mean trimmed regions.

Now, we are ready to formulate the key theoretical result of this Section, which formalizes the relation between coherent distortion risk measures and their uncertainty sets on one side, and weighted-mean trimmed regions on the other side. This is stated in the following Theorem 5.2, which is proved in Appendix 5.7. Also in the appendix, the geometrical properties of WM regions leading to such a relation are considered.

Theorem 5.2. *If* $\tilde{\mathbf{a}}$ *has an empirical distribution on* $\mathbf{a}^1, \ldots, \mathbf{a}^n$ *and* ρ *is a coherent distortion risk measure, then it holds:*

$$\{\mathbf{x} \in \mathbb{R}^d \,:\, \rho(\tilde{\mathbf{a}}'\mathbf{x} - b) \leq 0\} = \{\mathbf{x} \in \mathbb{R}^d \,:\, \mathbf{a}'\mathbf{x} \geq b \;\forall \mathbf{a} \in D_{\mathbf{w}_\alpha}(\mathbf{a}^1, \ldots, \mathbf{a}^n)\}. \tag{5.12}$$

The reader can see, that, loosely speaking, Theorem 5.2 provides a transition from a well-interpreted but hardly manageable risk constraint to an equivalent well-manageable constraint employing the geometrical construction of trimmed regions. In fact, recall that $D_{\mathbf{w}_\alpha}(\mathbf{a}^1, \ldots, \mathbf{a}^n)$ is a d-dimensional *convex polytope*, and thus the convex hull of a finite number of points (its vertices) or, equivalently, a bounded nonempty intersection of a finite number of closed halfspaces (that contain its facets). By this the calculation and representation of such a polytope can be done in two ways: either by its vertices or by its facets. Recall that a nonempty intersection of the polytope's boundary with a hyperplane is a *facet* if it has an affine dimension $d-1$, and a *ridge* if it has an affine dimension $d-2$. It is called an *edge* if it is a line segment, and a *vertex* if it is a single point. In general, each facet of a polytope in $\mathbb{R}^d$ is itself a polytope of dimension $d-1$ and has at least d vertices. With WM regions the number of a facet's vertices can vary considerably; it ranges between d and $d!$ (Bazovkin and Mosler, 2012a). That is why in calculating WM regions a representation by facets is preferable. In the following Section 5.3, we consider the topic in detail.

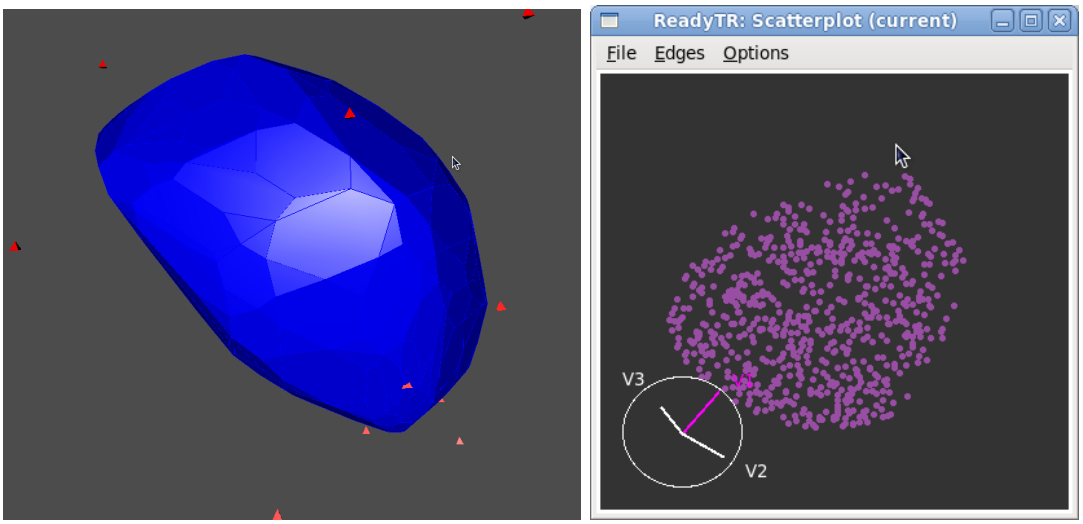

FIGURE 5.1: Visualization of WM regions by the *R* package WMTregions. Left panel: Facets of a three-dimensional region in $\mathbb{R}^3$. Right panel: Vertices of a four-dimensional region projected on a subspace of $\mathbb{R}^3$.

5.3 Solving the SLP with distortion risk constraint

5.3.1 Calculating the uncertainty set

In the previous section we have shown that the uncertainty set $\mathcal{U}_\rho$ equals the weighted-mean trimmed region $D_{\mathbf{w}_\alpha}$ for a properly chosen weight vector $\mathbf{w}_\alpha$. Bazovkin and Mosler (2012a) provide an algorithm by which this WM region can be exactly calculated in any dimension d.

The results can be visualized in dimensions two and three; for examples, see Figure 5.1.

It has been already mentioned in Chapter 3 that the number of vertices of a facet can be as much as $d!$. Therefore the representation of a WM region by its vertices appears to be less efficient than that by its facets. In the sequel, we will use the facet representation for solving the SLP.

5.3.2 The robust linear program

Using the result of Theorem 5.2, we can write down the robust linear program (5.1) with a distortion risk constraint in such a form:

$$\mathbf{c}'\mathbf{x} \longrightarrow \min \quad s.t.\ \mathbf{a}'\mathbf{x} \geq b \ \text{ for all } \mathbf{a} \in \mathcal{U}\,, \tag{5.13}$$

where the subscript ρ has been dropped for convenience. In simple words, we get a deterministic linear program with a set of constraints whose coefficients are contained

in a set $\mathcal{U}$ that is, according to Theorem 5.2, a weighted-mean trimmed region. The restriction in (5.13) is then rewritten as

$$\mathbf{x} \in \mathcal{X} = \bigcap_{\mathbf{a}\in\mathcal{U}} X_{\mathbf{a}}\,, \quad X_{\mathbf{a}} = \{\mathbf{x} : \mathbf{a}'\mathbf{x} \geq b\}\,. \tag{5.14}$$

Note that $\mathcal{X}$, as an intersection of a finite number of halfspaces, is a convex polyhedron. Therefore, a linear goal function is to be minimized on a convex polyhedron. Obviously, any optimal solution will lie on the boundary of $\mathcal{X}$.

5.3.3 Finding the optimum on the uncertainty set

In constructing an algorithm for the robust linear program, we explore the set $\mathcal{X}$ of feasible solutions and relate it to the uncertainty set $\mathcal{U}$ in the parameter space. It is shown that the space of solutions $\mathbf{x}$ and the space of coefficients $\mathbf{a}$ are, in some sense, *dual* to each other. The following two lemmas provide the connection between $\mathcal{X}$ and $\mathcal{U}$. First, we demonstrate that $\mathcal{X}$ is the intersection of those halfspaces whose normals are extreme points of $\mathcal{U}$.

Lemma 5.3. *It holds that*

$$\mathcal{X} = \bigcap_{\mathbf{a}\in\mathcal{U}} \{\mathbf{x} : \mathbf{a}'\mathbf{x} \geq b\} = \bigcap_{\mathbf{a}\in \operatorname{ext}\mathcal{U}} \{\mathbf{x} : \mathbf{a}'\mathbf{x} \geq b\}\,.$$

Proof. We show that $\bigcap_{\mathbf{a}\in \operatorname{ext}\mathcal{U}} X_{\mathbf{a}} \subset X_{\mathbf{u}}$ for all $\mathbf{u} \in \mathcal{U}$; then $\bigcap_{\mathbf{a}\in \operatorname{ext}\mathcal{U}} X_{\mathbf{a}} \subset \bigcap_{\mathbf{a}\in\mathcal{U}} X_{\mathbf{a}}$. The opposite inclusion is obvious. Assume $\mathbf{u} \in \mathcal{U}$. Then, as $\mathcal{U}$ is convex and compact, $\mathbf{u}$ is a convex combination of some points $\mathbf{a}^1, \ldots, \mathbf{a}^\ell \in \operatorname{ext}\mathcal{U}$, i.e., $\mathbf{u} = \sum_{i=1}^{\ell} \lambda_i \mathbf{a}^i$ with $\lambda_i \geq 0$ and $\sum_{i=1}^{\ell} \lambda_i = 1$, and for any $\mathbf{x} \in \bigcap_{\mathbf{a}\in \operatorname{ext}\mathcal{U}} X_{\mathbf{a}}$ holds $\mathbf{x} \in X_{\mathbf{a}^j}$ and ${\mathbf{a}^j}'\mathbf{x} \geq b$ for all j, hence $\mathbf{u}'\mathbf{x} = \sum_{i=1}^{\ell} \lambda_i {\mathbf{a}^i}'\mathbf{x} \geq b$, that is, $\mathbf{x} \in X_{\mathbf{u}}$. □

Lemma 5.3 says that each facet of the set $\mathcal{X}$ of feasible solutions corresponds to a vertex of the uncertainty set $\mathcal{U}$. Hence it is sufficient to consider the extreme points of the uncertainty set.

As a generalization of Lemma 5.3, we may prove by recursion on k: Each k-dimensional face of the feasible set corresponds to a $(d-k)$-dimensional face of the uncertainty set in the solution space. This resembles the dual correspondence between convex sets and their *polars* (cf. Rockafellar (1997)). However, in contrast to polars, this correspondence of facets is not reflexive.

From Lemma 5.3 it is immediately seen, how the robust optimization problem contrasts with a deterministic problem, where the empirical distribution of $\tilde{\mathbf{a}}$ concentrates at some

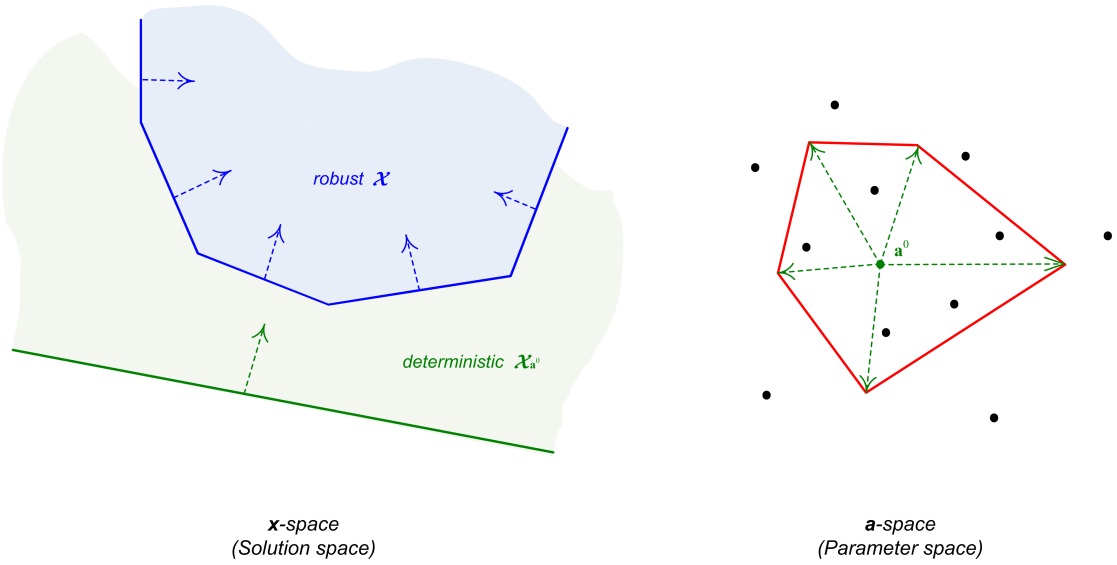

FIGURE 5.2: Deterministic and robust cases: feasible set (left panel), uncertainty set (right panel).

$\mathbf{a}^0 \in \mathcal{U}$. Observe that the deterministic feasible set $\mathcal{X}$ is just a halfspace, $\mathcal{X}_{\mathbf{a}^0} = \{\mathbf{x} : \mathbf{x}'\mathbf{a}^0 \geq b\}$. In the general robust case a halfspace is obtained for each $\mathbf{a} \in \operatorname{ext}\mathcal{U}$, and the robust feasible set $\mathcal{X}$ is their intersection. The halfspaces are bounded by hyperplanes with normals equal to $\mathbf{a} \in \operatorname{ext}\mathcal{U}$, and their intercepts are all the same and equal to b. Consequently, the robust feasible set $\mathcal{X}$ is always included in the deterministic feasible set $\mathcal{X}_{\mathbf{a}^0}$,

$$\mathcal{X} \subseteq \mathcal{X}_{\mathbf{a}^0} \quad \text{for any } \mathbf{a}^0 \in \mathcal{U}\,.$$

Moreover, the two feasible sets cannot be equal unless each element of $\mathcal{U}$ is a scalar multiple of $\mathbf{a}^0$ with a factor greater than one, $\mathcal{U} \subseteq \{\mathbf{a} : \mathbf{a} = \lambda\mathbf{a}^0, \lambda > 1\}$. Consequently, the minimum value of the robust stochastic LP cannot be smaller than the value of an LP with any deterministic parameter $\mathbf{a}^0$ chosen from the uncertainty set. Figure 5.2 (left panel) illustrates how a deterministic feasible set in dimension two compares to a general robust one: The line that bounds the halfspace $\mathcal{X}_{\mathbf{a}^0}$ 'folds' into a piecewise linear curve delimiting $\mathcal{X}$. In turn, Figure 5.2 (right panel) demonstrates the same relation between the uncertainty sets in the parameter space: the deterministic uncertainty set, which is a singleton $\mathbf{a}^0$, 'enlarges' into a non-degenerate uncertainty set containing $\mathbf{a}^0$.

Let

$$U_{\mathbf{x}} = \{\mathbf{a} \in \mathbb{R}^d : \mathbf{a}'\mathbf{x} \geq b\}\,, \quad \mathbf{x} \in \mathbb{R}^d\,.$$

Lemma 5.4. *It holds that*

$$\mathcal{U} \subset \bigcap_{\mathbf{x}\in\mathcal{X}} U_{\mathbf{x}} \subset \bigcap_{\mathbf{x}\in \operatorname{ext}\mathcal{X}} U_{\mathbf{x}}\,.$$

Moreover, each vertex $\mathbf{x} \in \operatorname{ext}\mathcal{X}$ *corresponds to a facet of* $\mathcal{U}$.

Proof. By Lemma 5.3 we have $\mathbf{x} \in \mathcal{X} \quad \Leftrightarrow \quad \mathbf{a}'\mathbf{x} \geq b$ for all $\mathbf{a} \in \mathcal{U}$. Now let $\mathbf{a} \in \mathcal{U}$; then for any $\mathbf{x} \in \mathcal{X}$ it holds that $\mathbf{a}'\mathbf{x} \geq b$, hence $\mathbf{a} \in U_{\mathbf{x}}$. Conclude $\mathcal{U} \subset \bigcap_{\mathbf{x}\in\mathcal{X}} U_{\mathbf{x}}$. Further, it is clear that an extreme point $\mathbf{x} \in \operatorname{ext}\mathcal{X}$ yields a facet of $\mathcal{U}$. □ □

Remark. While $\mathcal{U}$ is always compact, $\mathcal{X}$ is in general not. Therefore neither inclusion holds with equality.

According to Lemma 5.3, we could now reformulate the robust LP (5.13), basing it on the constraints generated by the vertices of $\mathcal{U}$:

$$\mathbf{c}'\mathbf{x} \longrightarrow \min \quad s.t.\ \mathbf{a}'\mathbf{x} \geq b \ \text{ for all } \mathbf{a} \in \operatorname{ext}\mathcal{U}\,,$$

and then apply the ordinary simplex method to it. However, usually WM regions have a very large number of vertices, because even a single facet can have up to $d!$ ones. Bazovkin and Mosler (2012a) have shown this number to lie between $\mathcal{O}(n^d)$ and $\mathcal{O}(\frac{n^{2d}}{2^d})$ depending on the type of the WM region for the data cloud of n points, thus, obviously, making the basic straightforward approach almost inapplicable here. From the other side, calculated WM regions are efficiently represented by their facets. In our algorithm, we pursue another way to find the optimal solution, namely searching it even without explicit construction of $\mathcal{X}$ and taking advantage of the facets representation of $\mathcal{U}$.

To manage this task let us consider the goal function $\mathbf{c}'\mathbf{x}$. In the solution space the optimization vector $\mathbf{c}$ defines a direction, which can be also determined by a set of hyperplanes orthogonal to this direction. Clearly, all these hyperplanes are parallel and their normals are some multiples of $\mathbf{c}$. For example, in dimension two for $\mathbf{c} = (2.1, 1.4)'$ and $b = 5$ the hyperplanes $\{\mathbf{x} : (2.1, 1.4)' \cdot \mathbf{x} = 5\}$ and $\{\mathbf{x} : (4.2, 2.8)' \cdot \mathbf{x} = 5\}$ belong to such set. Recall that we have fixed the intercept at the value of b for all the hyperplanes in the solution space, and can differentiate them only by their normals. In the parameter space, each of these hyperplanes corresponds to the point $\mathbf{c}$ multiplied with the relevant scaling factor. Hence the image of all the hyperplanes in the parameter space is obtained by moving a point along a straight ray φ that starts at the origin and contains $\mathbf{c}$, as it is shown on Figure 5.3.

One of the hyperplanes touches $\mathcal{X}$ at the optimum. All others are either intersecting the interior of $\mathcal{X}$ or not intersecting it at all. This means that the touching hyperplane corresponds to a point lying on the surface of $\mathcal{U}$. Therefore, we should search the intersection of $\mathcal{U}$ with the ray φ, and, because of Lemma 5.4, the intersected facet of $\mathcal{U}$ corresponds to the optimal vertex of $\mathcal{X}$.

Note that finding the intersection of a line and a polyhedron in $\mathbb{R}^3$ is an important problem in computer graphics (cf. Kay and Kajiya (1986)). The same principle is

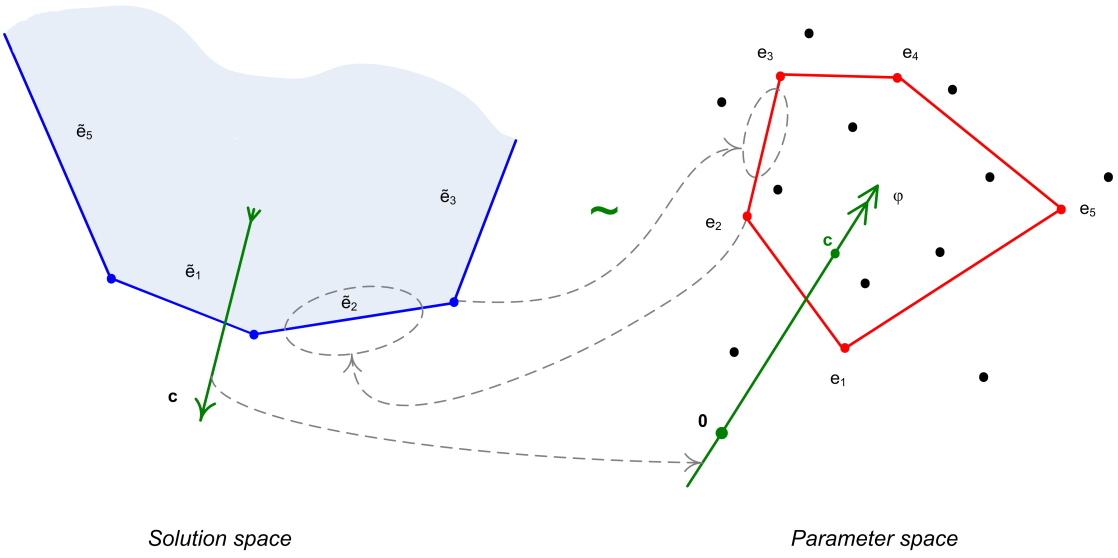

FIGURE 5.3: Duality between spaces.

employed for a general dimension d. The uncertainty set $\mathcal{U}$ is the finite intersection of halfspaces $\mathcal{H}_j$, $j = 1 \dots J$, each being defined by a hyperplane H_j with normal $\mathbf{n}_j$ pointing into $\mathcal{H}_j$ and an intercept d_j.

Consider some point $\mathbf{u}$ on the ray φ that is not in $\mathcal{U}$. Compute $\frac{d_j}{\mathbf{u}'\mathbf{n}_j}$ for all halfspaces $\mathcal{H}_j$ that do *not* include $\mathbf{u}$, i.e., where $(\mathbf{u}'\mathbf{n}_j - d_j) < 0$ holds. (In other words, H_j is *visible* from $\mathbf{u}$.) Find j_* at which this value is the largest. Recall that moving a point $\mathbf{u}$ along φ is equivalent to multiplying $\mathbf{u}$ by some constant. The furthest move is given by the biggest constant. The *optimal solution* $\mathbf{x}^*$ of the robust SLP has to satisfy $\mathbf{a}'\mathbf{x}^* \geq b$, which is equivalent to

$$\mathbf{a}' \left(\frac{d_{j_*}}{b} \mathbf{x}^* \right) \geq d_{j_*} .$$

Hence, to obtain $\mathbf{x}^*$, the normal $\mathbf{n}_{j_*}$ has to be scaled with the constant $\frac{b}{d_j}$,

$$\mathbf{x}^* = \frac{b}{d_{j_*}} \mathbf{n}_{j_*} . \tag{5.15}$$

Besides the regular situation described above, two special cases can arise:

1. There is no facet visible from the origin. This means that no solution is obtained.
2. φ does not intersect $\mathcal{U}$. Then the whole procedure is repeated with the opposite ray $-\varphi$. If this still gives no intersection, an infinite solution exists.

Finally, we should point out that not the whole polytope $\mathcal{U}$ needs to be calculated but only that part of it that intersects the ray φ. In searching for the optimum not all F facets need to be checked, but only a subset of the surface where the intersection will

happen. Such a filtration makes the procedure more efficient. Next, we show how to select this subset.

Let $\mathbf{x}^*$ be an *optimal solution* of the robust SLP. A subset $\mathcal{U}_{\text{eff}}$ of $\mathcal{U}$ will be mentioned as an *efficient parameter set* if

- $\mathbf{x}^*$ remains the solution for $\bigcap_{\mathbf{a}\in\mathcal{U}_{\text{eff}}}\{\mathbf{x} : \mathbf{a}'\mathbf{x} \geq b\ \} \supset \mathcal{X}$ and
- $\mathbf{a}, \mathbf{d} \in \mathcal{U}_{\text{eff}},\ \mathbf{a}'\mathbf{x} \geq b \Rightarrow \mathbf{d}'\mathbf{x} \geq b,\ \forall \mathbf{x}$ implies $\mathbf{a} = \mathbf{d}$.

That is to say, $\mathcal{U}_{\text{eff}}$ is the minimal subset of $\mathcal{U}$ containing all facets that can be optimal for some $\mathbf{c}$.

Proposition 5.5. *$\mathcal{U}_{\text{eff}}$ is the union of all facets of $\mathcal{U}$ for which $d_j \geq 0$ holds.*

In other words, an efficient parameter set $\mathcal{U}_{\text{eff}}$ consists of that part of the surface of $\mathcal{U}$ that is visible from the origin $\mathbf{0}$. The proof is obvious.

To visualize the efficient parameter set we use the *augmented uncertainty set*, which is defined as

$$\{\mathbf{a} : \mathbf{a} = \lambda\mathbf{a}^*, \lambda > 1, \mathbf{a}^* \in \mathcal{U}_{\text{eff}}\}\,.$$

It includes all parameters that are dominated by $\mathcal{U}_{\text{eff}}$; see the shaded area in the right panel of Figure 5.4.

So far we have assumed that $b > 0$. It is easy to show, that with $b < 0$ we have to construct the intersection of φ with the part of the surface of $\mathcal{U}$ that is *invisible* from the origin $\mathbf{0}$, which is $\tilde{\mathcal{U}}_{\text{eff}}$ in this case. In the sense of Proposition 5.5, $\tilde{\mathcal{U}}_{\text{eff}}$ contains all facets of $\mathcal{U}$ with $d_j \leq 0$. Obviously, $\tilde{\mathcal{U}}_{\text{eff}}$ is always non-empty in this case, which, in turn, means that the existence of a solution is guaranteed. However, the solution can be infinite if φ does not intersect $\tilde{\mathcal{U}}_{\text{eff}}$.

The situation of $b < 0$ is common in the maximizing SLPs. In fact, if we have the model

$$\mathbf{c}'\mathbf{x} \longrightarrow \max \quad s.t.\ \mathbf{a}'\mathbf{x} \leq b \ \text{ for all } \mathbf{a} \in \mathcal{U}\,, \tag{5.16}$$

it is possible to rewrite it as follows:

$$(-\mathbf{c})'\mathbf{x} \longrightarrow \min \quad s.t.\ (-\mathbf{a})'\mathbf{x} \geq -b \ \text{ for all } \mathbf{a} \in \mathcal{U}\,. \tag{5.17}$$

Clearly, (5.17) is equivalent to (5.13) except for the negativity of the coefficient b.

5.4 The algorithm

In this part an accurate procedure for obtaining the optimal solution is given.

Input

- a vector $\mathbf{c} \in \mathbb{R}^d$ of coefficients of the goal function,
- n observations $\{\mathbf{a}^1, \ldots, \mathbf{a}^n\} \subset \mathbb{R}^d$ of coefficient vectors of the restriction,
- a right-hand side $b \in \mathbb{R}$ of the restriction,
- a distortion risk measure ρ (defined either by name or by a weight vector).

Output

- the uncertainty set $\mathcal{U}$ of parameters given by
  - facets (i.e., normals and intercepts),
  - vertices,
- the optimal solution $\mathbf{x}^*$ of the robust LP and its value $\mathbf{c}'\mathbf{x}^*$.

Steps of the Algorithm

A. Calculate the subset $\mathcal{U}_{\text{eff}} \subset \mathcal{U}$ consisting of facets $\{(\mathbf{n}_j, d_j)\}_{j \in J}$.

B. Create a line φ passing through the origin $\mathbf{0}$ and $\mathbf{c}$.

C. Search for a facet H_{j_*} of $\mathcal{U}_{\text{eff}}$ that is intersected by φ:

   a. Select a subset $\mathcal{U}_{sel} \subseteq \mathcal{U}_{\text{eff}}$ of facets: This may be either $\mathcal{U}_{\text{eff}}$ itself or its part where the intersection is expected; $\mathcal{U}_{sel} = \{(\mathbf{n}_j, d_j) : j \in J_{sel}\}$. For example, we can search for the best solution on a pre-given subset of parameters. The other possible filtration is iterative transition to a facet with better criterion value.

   b. Take a point $\mathbf{u} = \lambda\mathbf{c}, \lambda \geq 0$, outside the augmented uncertainty set. Find the $j_* = \arg\max_j \{\lambda_j = \frac{d_j}{\mathbf{u}'\mathbf{n}_j} : \lambda_j > 0\}_{j \in J_{sel} \subseteq J}$. For the case $b < 0$ just replace $\arg\max$ with $\arg\min$.

      I. If φ does not intersect $\mathcal{U}_{\text{eff}}$, then the solution is *infinite*. If $b > 0$, then repeat C.b. for the opposite ray $-\varphi$; else stop.

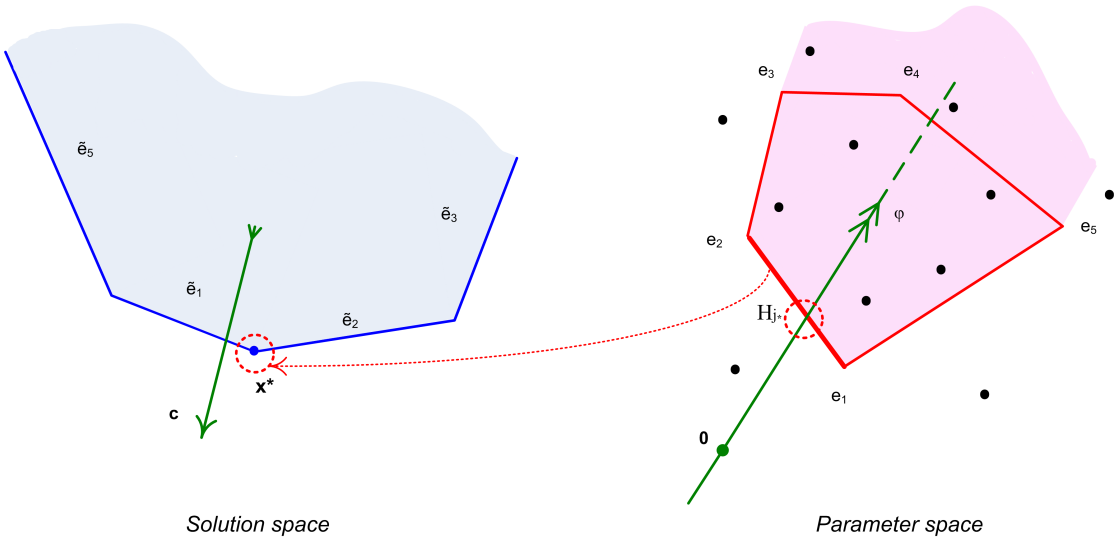

FIGURE 5.4: Finding the optimal solution on the uncertainty set.

II. If in the case $b > 0$ the inner part of $\mathcal{U}$ contains the origin, then *no solution* exists; stop.

c. $\mathbf{x}^* = \frac{b}{d_{j*}}\mathbf{n}_{j*}$ is the optimal solution of the robust LP.

In fact, the line φ consists of points that correspond to hyperplanes whose normal is the vector $\mathbf{c}$ in the dual space. One part of φ is dominated by points from $\mathcal{U}_{\text{eff}}$, while the other is not (which results from Proposition 5.5). The crossing point $\mathbf{a}^*$ defines the hyperplane that touches the feasible set at the optimum as its dual.

Moreover, a typical nonnegativity side constraint $\mathbf{x} \geq \mathbf{0}$ can easily be accounted for in the algorithm. In considering this, the search for facets has just to be restricted to those having nonnegative normals.

To solve the portfolio selection problem (5.6) with the algorithm, we treat the realizations of the vector $-\tilde{\mathbf{r}}$ of losses rates as $\{\mathbf{a}^1, \ldots, \mathbf{a}^n\}$, and minimize $\mathbf{c}'\mathbf{x}$ with $\mathbf{c} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{a}^i$. This corresponds to transforming the maximizing SLP by (5.17) and running the procedure outlined above. Note that both φ and $\mathcal{U}$ contain the point $\frac{1}{n}\sum_{i=1}^{n}\mathbf{a}^i$, that is, they always intersect, which, in turn, guarantees the existence of a finite solution. To meet a unit budget constraint, the solution $\mathbf{x}^*$ is finally scaled down by $\sum_{j=1}^{d} x_j^* = 1$. Recall that the risk measure is, by definition, scale equivariant.

5.4.1 Sensitivity and complexity issues

Next, we discuss how the robust SLP and its optimal solution behave when the data $\{\mathbf{a}^1, \ldots, \mathbf{a}^n\}$ on the coefficients are slightly changed. From (5.18) it is immediately seen that the support function $h_{\mathcal{U}}$ of the uncertainty set is continuous in the data $\mathbf{a}^j$ as

well as in the weight vector $\mathbf{w}_\alpha$. (Note that the support function $h_{\mathcal{U}}$ is even *uniformly continuous* in $\mathbf{a}^1, \ldots, \mathbf{a}^n$ and $\mathbf{w}_\alpha$, which is tantamount saying that the uncertainty set $\mathcal{U}$ is *Hausdorff continuous* in the data and the risk weights.) Consequently, a slight perturbation of the data will only slightly change the value of the support function of $\mathcal{U}$, which is a practically useful result regarding the sensitivity of the uncertainty set with respect to the data. The same is true for a small change in the weights of the risk measure.

We conclude that the point $\mathbf{a}^{j*}$ where the line through the origin and $\mathbf{c}$ cuts $\mathcal{U}$ depends continuously on the data and the weights. However this is not true for the optimal solution $\mathbf{x}^*$, which may 'jump' when the cutting point moves from one facet of $\mathcal{U}$ to a neighboring one.

The theoretical complexity in time of finding the solution is compounded from the complexity of one transition to the next facet and by the whole number of such transitions until the sought-for facet is achieved. Bazovkin and Mosler (2012a) have shown that the transition has a complexity of $\mathcal{O}(d^2 n)$. In turn, in the same paper the number of facets $N(n, d)$ of an WM region is shown to lie between $\mathcal{O}(n^d)$ and $\mathcal{O}(n^{2d})$ depending on the type of the WM region. Thus, it is easily seen, that an average number of facets in a facets chain of a fixed length is defined by the density of facets on the region's surface, $\sqrt[d]{N(n,d)}$, and is estimated by a function between $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$. The overall complexity is then $\mathcal{O}(d^2 n^2)$ up to $\mathcal{O}(d^2 n^3)$. Notice, that the lower complexity is achieved for zonoid regions, namely when the expected shortfall is used for the risk measure.

5.4.2 Ordered sensitivity analysis

Alternative uncertainty sets that are ordered by inclusion can be also compared. From Lemma 5.3 it is clear that the respective sets of feasible solutions are then ordered in the reverse direction; see, e.g., Figure 5.5. In particular we can consider the robust LP for two alternative distortion risk measures based on weight vectors $\mathbf{w}_\alpha$ and $\mathbf{w}_\beta$, respectively, that satisfy the monotonicity restriction (5.11). Then the resulting uncertainty sets are nested, $\mathcal{U}_\beta \subset \mathcal{U}_\alpha$ and so are, conversely, the feasible sets, $\mathcal{X}_\beta \supset \mathcal{X}_\alpha$. This is a useful approach for visualizing the sensitivity of the robust LP against changes in risk evaluation.

5.5 Robust SLP for generally distributed coefficients

So far an SLP (5.1) has been considered where the coefficient vector $\tilde{\mathbf{a}}$ follows an empirical distribution. It has been solved on the basis of n observations $\{\mathbf{a}^1, \ldots, \mathbf{a}^n\}$. In this

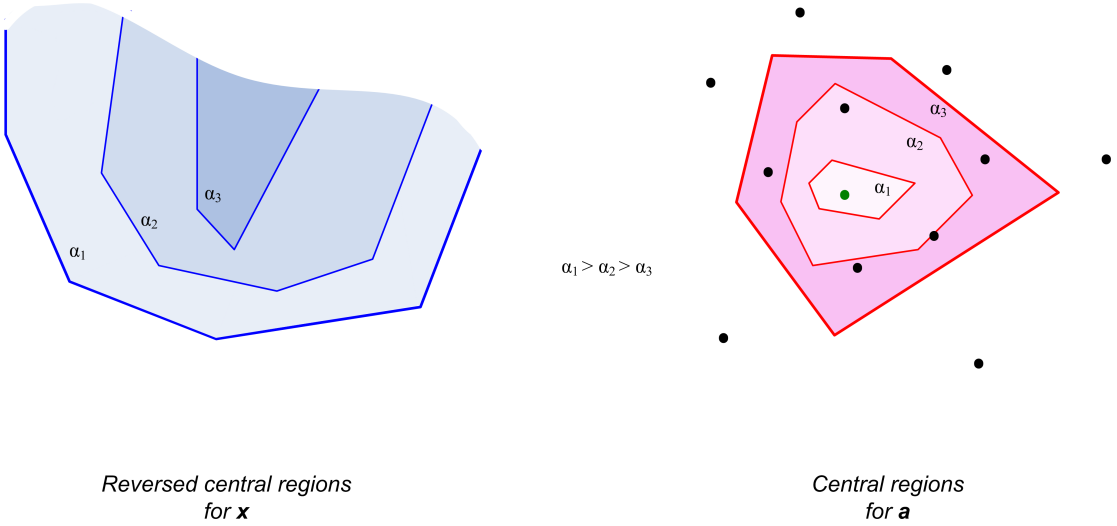

FIGURE 5.5: Example of the 'reversed' central regions in the dimension 2.

section the SLP is addressed with a general probability distribution P of $\tilde{\mathbf{a}}$. We formulate the robust SLP in the general case and demonstrate that the solution of this SLP can be consistently estimated by random sampling from P.

Consider a distortion risk measure ρ (5.8) that measures the risk of a general random variable Y and has weight generating function r, $\rho(Y) = -\int_0^1 Q_Y(t)dr(t)$. As in Section 5.2.2 a convex compact $\mathcal{U}$ in $\mathbb{R}^d$ is constructed through its support function $h_{\mathcal{U}}$,

$$h_{\mathcal{U}}(\mathbf{p}) = \int_0^1 Q_{\mathbf{p}'\tilde{\mathbf{a}}}(t)dr(t)\,.$$

Now, let a sequence $(\tilde{\mathbf{a}}^n)_{n\in\mathbb{N}}$ of independent random vectors be given that are identically distributed with P, and consider the sequence of random uncertainty sets $\mathcal{U}_n$ based on $\tilde{\mathbf{a}}^1, \ldots, \tilde{\mathbf{a}}^n$. Dyckerhoff and Mosler (2011) have shown:

Proposition 5.6 (Dyckerhoff and Mosler (2011))**.** *$\mathcal{U}_n$ converges to $\mathcal{U}$ almost surely in the Hausdorff sense.*

The proposition implies that by drawing an independent sample of $\tilde{\mathbf{a}}$ and solving the robust LP based on the observed empirical distribution a consistent estimate of the uncertainty set $\mathcal{U}$ is obtained. Moreover, the cutting point $\mathbf{a}^{j*}$, where the line through the origin and $\mathbf{c}$ hits the uncertainty set, is consistently estimated by our algorithm. If an ambiguous solution is possible, in particular for a discretely distributed $\tilde{\mathbf{a}}$, the algorithm calculates one of the available solutions consistently. In fact, the optimal solution $\mathbf{x}^*$

may perform a jump when $\mathbf{a}^{j*}$ moves from one facet of $\mathcal{U}$ to a neighboring one, however the algorithm for determining $\mathbf{x}^*$ selects always a unique facet containing $\mathbf{a}^{j*}$.

5.6 Concluding remarks

A stochastic linear program (SLP) has been investigated, where the coefficients of the linear restrictions are random. Distortion risk constraints are imposed on the random restrictions and an equivalent robust SLP is modeled, whose worst-case solution is searched over an uncertainty set of coefficients. If the risk is measured by a general coherent distortion risk measure, the uncertainty set of a restriction has been shown to be a *weighted-mean trimmed region*. This provides a comprehensive visual and computable characterization of the uncertainty set. An *algorithm* has been developed that solves the robust SLP under a single stochastic constraint, given a set of observations. It is available as an *R* package StochaTR (Bazovkin and Mosler, 2012b). Moreover, if the data is generated by an infinite i.i.d. sample, the limit behavior of the solution has been investigated. The algorithm allows the introduction of additional deterministic constraints, in particular, those regarding nonnegativity.

TABLE 5.1: Running times of StochaTR for different n and d (in seconds).

$d\backslash n$	1000	2000	3000	4000	5000	10000	15000	20000	25000
3	0.3	1.14	1.76	2.92	3.41	6.18	12.61	15.06	47.54
4	0.66	2.21	3.47	4.48	4.27	7.68	16.97	20.04	
5	1.85	3.09	5.68	9.28	11.03	13.52	27.34	54.86	
6	2.08	4.41	5.62	14.99	18.73	25.07	46.88		
7	2.16	6.22	13.3	25.44	28.56	52.33			
8	4.18	9.78	20.18	31.82	34.23				
9	5.18	14.75	24.11	35.94	61.14				
10	6.17	16.97	33.82	42.11	67.06				

Table 5.1 reports simulated running times (in seconds) of the *R* package for the 5%-level expected shortfall and different d and n. The data are simulated by mixing the uniform distribution on a d-dimensional parallelogram with a multivariate Gaussian distribution. In light of the table the complexity seems to grow with d and n slower than $\mathcal{O}(d^2n^2)$.

Besides this, we contrast our new procedure with the seminal approach of Rockafellar and Uryasev (2000), who solve the portfolio problem by optimizing the expected shortfall with a simplex-based method. In illustrating their method, they simulate three-dimensional normal returns having specified expectations and covariance matrices. We have applied our package to likewise simulated data on a 1.73GHz single-core CPU with at most 1.5 gigabytes of memory available. The computational times are exhibited in Table 5.2. For a comparison, some cells also contain a second value that

corresponds to the Rockafellar and Uryasev (2000) procedure and is taken from Table 5 there. Our algorithm usually needs some dozens of iterations only, which is substantially fewer than the algorithm of Rockafellar and Uryasev (2000). Also, in contrast to the latter, where the resulting portfolio can vary between $(0.42, 0.13, 0.45)$ for $n = 1000$ and $(0.64, 0.04, 0.32)$ for $n = 5000$, we get a *stable* optimal portfolio. Our solution averages at $(0.36, 0.15, 0.49)$, which has approximately the same V@R and expected shortfall as that in the compared study but yields a *better* value of the *expected return*. Note that the computational times reported in Rockafellar and Uryasev (2000) do not differ much from ours.

TABLE 5.2: Running times of StochaTR for different n and α (in seconds); in parentheses running times of Rockafellar and Uryasev (2000).

$\alpha\backslash n$	1000	5000	10000	15000	20000	25000
0.10	1.1 (<5)	7.2 (6)	23.7 (20)	46	56.3 (45)	74.4
0.05	0.5 (<5)	4.7 (6)	14.0 (12)	20.0	39.8 (40)	53.2
0.01	0.3 (<5)	2.3 (6)	3.8 (6)	7.9	22.1 (50)	38.5

Finally, our approach turns out to be very flexible. In particular, non-sample information can be introduced into the procedure in an interactive way by explicitly changing and modifying the uncertainty set. A possibility of extending the algorithm to solve SLPs with multiple constraints (5.2) will be shown in Chapter 6. Also procedures that allow for a stochastic right-hand side in the constraints and random coefficients in the goal function will be explored in Chapter 6.

5.7 Appendix: Technical details

In this supplement, we first discuss the principal properties of distortion risk measures and the characterization of a risk bound by an uncertainty set. Then we describe WM regions fully by their projections on lines. Based on these notions and facets, next, Theorem 5.2 is proved. Finally, the coherency property of WM regions is demonstrated.

5.7.1 Properties of distortion risk measures

Let us consider a probability space $\langle \Omega, \mathcal{F}, P \rangle$ and a set $\mathcal{R}$ of random variables (e.g., returns of portfolios). A function $\rho : \mathcal{R} \to \mathbb{R}$ is a monetary *risk measure* if for $Y, Z \in \mathcal{R}$ it holds:

1. *Monotonicity*: If Y is pointwise larger than Z, $Y \geq Z$, then it has less risk, $\rho(Y) \leq \rho(Z)$.

2. *Translation invariance*: $\rho(Y+\gamma) = \rho(Y) - \gamma$ for all $\gamma \in \mathbb{R}$.

   A risk measure is law-invariant if it holds additionally:

3. *Law-invariance*: If Y and Z have the same distribution, $P_Y = P_Z$, then $\rho(Y) = \rho(Z)$.

A law-invariant risk measure ρ is *coherent* if it is, in addition, positive homogeneous and subadditive,

4. *Positive homogeneity*: $\rho(\lambda Y) = \lambda\rho(Y)$ for all $\lambda \geq 0$,

5. *Subadditivity*: $\rho(Y+Z) \leq \rho(Y) + \rho(Z)$ for all $Y, Z \in \mathcal{R}$.

The last two restrictions imply that *diversification* is encouraged by the risk measure - a crucial property in risk management. For the theory of risk measures, see, e.g., Föllmer and Schied (2004). Loosely speaking, diversification is a natural mechanism of reducing risk by 'not putting all the eggs into one basket'.

Note that a distortion risk measure (5.8) satisfies the above properties 1 to 3, hence is a law-invariant risk measure.

A function $\rho : \mathcal{R} \to \mathbb{R}$ is said to satisfy the *Fatou property* if for any bounded sequence converging pointwise to Y, $\liminf_{n\to\infty}\rho(Y_n) \geq \rho(Y)$ holds. With the notion of coherent risk measures, we reformulate a fundamental representation result of Huber (1981):

Proposition 5.7. *ρ is a coherent risk measure satisfying the Fatou property if and only if there exists a family $\mathbb{Q}$ of probability measures that are dominated by P (i.e., $P(S) = 0 \Rightarrow Q(S) = 0$ for any $S \in \mathcal{F}$ and $Q \in \mathbb{Q}$) such that for all $Y \in \mathcal{R}$*

$$\rho(Y) = \sup_{Q\in\mathbb{Q}} E_Q(-Y).$$

We say that the family $\mathbb{Q}$ generates ρ. In particular, let $(\Omega, \mathcal{A}) = (\mathbb{R}^d, \mathcal{B}^d)$ and P be the probability distribution of a random vector $\tilde{\mathbf{a}}$. Huber's Theorem implies that for any coherent risk measure ρ there exists a family $\mathbb{G}$ of P-dominated probabilities on $\mathcal{B}^d$ so that

$$\begin{aligned}
\rho(\tilde{\mathbf{a}}'\mathbf{x} - b) \leq 0 \quad &\Leftrightarrow \quad \rho(\tilde{\mathbf{a}}'\mathbf{x}) \leq -b \\
&\Leftrightarrow \quad \inf_{G\in\mathbb{G}} E_G(\tilde{\mathbf{a}}'\mathbf{x}) \geq b \\
&\Leftrightarrow \quad E_G(\tilde{\mathbf{a}}'\mathbf{x}) \geq b \text{ for all } G \in \mathbb{G}.
\end{aligned}$$

Let us denote the unit simplex in $\mathbb{R}^n$ by Δ^n,

$$\Delta^n = \{\mathbf{q} \in \mathbb{R}^n \, : \, \sum_{k=1}^{n} q_k = 1, q_k \geq 0 \;\; \forall k\} \, .$$

Then, if $\tilde{\mathbf{a}}$ has an empirical distribution on n given points in $\mathbb{R}^d$, any subset $\mathcal{Q}$ of Δ^n corresponds to a family of P-dominated probabilities, and thus defines a coherent risk measure ρ. As an immediate consequence of Huber's theorem an equivalent characterization of the risk constraint is obtained (see also Bertsimas and Brown (2009)):

Proposition 5.8. *Let $\rho : \mathcal{R} \to \mathbb{R}$ be a coherent risk measure and let $\tilde{\mathbf{a}}$ have an empirical distribution on $\mathbf{a}^1, \ldots, \mathbf{a}^n \in \mathbb{R}^d$. Then there exists some $\mathcal{Q}_\rho \subset \Delta^n$ such that*

$$\rho(\tilde{\mathbf{a}}'\mathbf{x} - b) \leq 0 \Leftrightarrow \mathbf{a}'\mathbf{x} \geq b \;\; \textit{for all}$$
$$\mathbf{a} \in \mathcal{U}_\rho := \operatorname{conv}\{\mathbf{a} \in \mathbb{R}^d \, : \, \mathbf{a} = \sum_{i=1}^{n} q_i \mathbf{a}^i, (q_1, \ldots, q_n) \in \mathcal{Q}_\rho\} \, .$$

Here, $\operatorname{conv}(W)$ denotes the convex closure of a set W. Proposition 5.8 says that a deterministic restriction $\mathbf{a}'\mathbf{x} \geq b$ holding uniformly for all $\mathbf{a}$ in the *uncertainty set* $\mathcal{U}_\rho$ is equivalent to the risk constraint (5.5) on the stochastic restriction.

5.7.2 Characterization of WM regions

A WM region (5.10) is characterized by its projections on lines. Note that each $\mathbf{p} \in S^{d-1}$, where S^{d-1} is a $(d-1)$-variate unit sphere, yields a projection of the data $\mathbf{a}^1, \ldots, \mathbf{a}^n$ on the line generated by $\mathbf{p}$ and thus induces a permutation $\pi_{\mathbf{p}}$ of the data,

$$\mathbf{p}'\mathbf{a}^{\pi_{\mathbf{p}}(1)} \leq \mathbf{p}'\mathbf{a}^{\pi_{\mathbf{p}}(2)} \leq \cdots \leq \mathbf{p}'\mathbf{a}^{\pi_{\mathbf{p}}(n)} \, .$$

The permutation is not necessarily unique - and let $H(\mathbf{a}^1, \ldots, \mathbf{a}^n)$ denote the set of all directions $\mathbf{p} \in S^{d-1}$ that induce a non-unique permutation $\pi_{\mathbf{p}}$. Recall that the *support function* of a closed convex set K is defined as $h(\mathbf{p}) = \sup\{\mathbf{p}'\mathbf{x} : \mathbf{x} \in K\}$, $\mathbf{p} \in S^{d-1}$. Given a convex polytope K, an *extreme point* of K is the unique solution of $\max\{\mathbf{p}'\mathbf{x} : \mathbf{x} \in K\}$ for some $\mathbf{p} \in S^{d-1}$. Dyckerhoff and Mosler (2011) have shown that the support function h_α of $D_{\mathbf{w}_\alpha} = D_{\mathbf{w}_\alpha}(\mathbf{a}^1, \ldots, \mathbf{a}^n)$ amounts to

$$h_\alpha(\mathbf{p}) = \sum_{j=1}^{n} w_{\alpha,j} \mathbf{p}'\mathbf{a}^{\pi_{\mathbf{p}}(j)} \, , \quad \mathbf{p} \in S^{d-1} \, . \tag{5.18}$$

It follows that, whenever $\pi_{\mathbf{p}}$ is unique, the polytope $D_{\mathbf{w}_\alpha}$ has an extreme point in direction $\mathbf{p}$, which is given by

$$\sum_{j=1}^{n} w_{\alpha,j}\mathbf{a}^{\pi_{\mathbf{p}}(j)} = \sum_{i=1}^{n} w_{\alpha,\pi_{\mathbf{p}}^{-1}(i)}\mathbf{a}^{i}\,, \quad \mathbf{p} \in S^{d-1} \setminus H(\mathbf{a}^1,\ldots,\mathbf{a}^n)\,. \tag{5.19}$$

5.7.3 Proof of Theorem 5.2

Now we are moving to the proof of Theorem 5.2. From (5.9) and (5.19) we can see that, with $q_i = w_{\alpha,\pi_{\mathbf{p}}^{-1}(i)}$ and $y_{[i]} = -\mathbf{p}'\mathbf{a}^i$, the extreme point of the projection of $D_{\mathbf{w}_\alpha}$ on the $\mathbf{p}$-line is obtained by applying a $\mathbf{q}$-distortion risk measure to the projected data points. Now, setting

$$\mathcal{Q}_\rho = \{\mathbf{q} \in \Delta^n \,:\, \mathbf{q} = (w_{\alpha,\pi_{\mathbf{p}}^{-1}(1)},\ldots w_{\alpha,\pi_{\mathbf{p}}^{-1}(n)}), \mathbf{p} \in S^{d-1} \setminus H(\mathbf{a}^1,\ldots,\mathbf{a}^n)\}$$

and having $\mathcal{U}_\rho = \operatorname{conv}\{\mathbf{a} \,:\, \mathbf{a} = \sum_{i=1}^n q_i\mathbf{a}^i, \mathbf{q} \in \mathcal{Q}_\rho\}$ according to Proposition 5.8, we get that all extreme points of $D_{\mathbf{w}_\alpha}$ are in $\mathcal{U}_\rho$, hence $D_{\mathbf{w}_\alpha} \subseteq \mathcal{U}_\rho$. On the other hand, for every $\mathbf{q} \in \mathcal{Q}_\rho$ it holds that $\sum_{i=1}^n q_i\mathbf{a}^i \in D_{\mathbf{w}_\alpha}$, which implies $\mathcal{U}_\rho \subseteq D_{\mathbf{w}_\alpha}$. We conclude that $\mathcal{U}_\rho = D_{\mathbf{w}_\alpha}$.

Thus, using the result from Proposition 5.8, we have proven the equality between the distortion risk constraint feasible set and a properly chosen WM region, which parallels Theorem 4.3 in Bertsimas and Brown (2009) and proves Theorem 5.2.

Finally, we would like to emphasize some facts that are used for constructing the trimmed regions as convex polytopes. The vertices of a polytope are its extreme points. From the above we know that the directions $\mathbf{p} \in S^{d-1} \setminus H(\mathbf{a}^1,\ldots,\mathbf{a}^n)$ belong to vertices, while the directions $\mathbf{p} \in H(\mathbf{a}^1,\ldots,\mathbf{a}^n)$ belong to parts of the boundary that have affine dimension ≥ 1.

5.7.4 Risk-relevant properties of WM regions

In the context of risk measurement it is crucial that the WM regions possess two properties that enable them to generate coherent risk measures: *monotonicity* and *subadditivity*.

Proposition 5.9. (Coherency properties of WM regions)

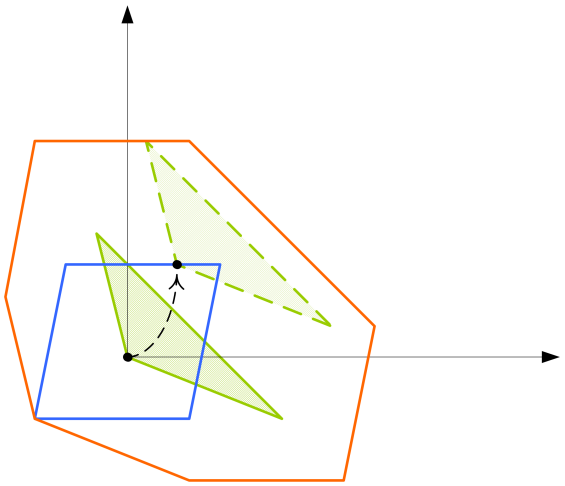

FIGURE 5.6: An illustration of the subadditivity property.

1. *Monotonicity: If* $\mathbf{z}_k \leq \mathbf{y}_k$ *holds for all* k *(in the componentwise ordering of* $\mathbb{R}^d$*), then*

$$D_{\mathbf{w}_\alpha}(\mathbf{y}_1, \ldots, \mathbf{y}_n) \subset D_{\mathbf{w}_\alpha}(\mathbf{z}_1, \ldots, \mathbf{z}_n) \oplus \mathbb{R}^d_+ , \quad \textit{and}$$
$$D_{\mathbf{w}_\alpha}(\mathbf{z}_1, \ldots, \mathbf{z}_n) \subset D_{\mathbf{w}_\alpha}(\mathbf{y}_1, \ldots, \mathbf{y}_n) \oplus \mathbb{R}^d_- .$$

2. *Subadditivity:*

$$D_{\mathbf{w}_\alpha}(\mathbf{y}_1 + \mathbf{z}_1, \ldots, \mathbf{y}_n + \mathbf{z}_n) \subset D_{\mathbf{w}_\alpha}(\mathbf{y}_1, \ldots, \mathbf{y}_n) \oplus D_{\mathbf{w}_\alpha}(\mathbf{z}_1, \ldots, \mathbf{z}_n) .$$

In this Proposition the symbol $\oplus$ is the *Minkowski addition*, $A \oplus B = \{a + b : a \in A, b \in B\}$ for A and $B \subset \mathbb{R}^d$. For a proof, see Dyckerhoff and Mosler (2011).

The subadditivity property of WM regions is an immediate extension of the subadditivity restriction usually imposed on univariate risk measures. In dimensions two and more it has an interpretation as a dilation of one trimmed region by the other. To understand this better let us consider the simple example of the Minkowski addition given in Figure 5.6. The figure exhibits a solid triangle with one vertex at the origin and a dotted-border quadrangle. Now move the triangle in such a way that its lower left corner passes all points of the quadrangle. At each point of the quadrangle we get a copy of the initial triangle (with a dashed border) shifted by the coordinate of the point. The union of all these triangles gives us the Minkowski sum of the initial two sets, which is the big heptagon in Figure 5.6. Observe that, if the rectangle is moved around the triangle, the same sum is obtained. The subadditivity states that if, e.g., these two figures are WM regions $D_{\mathbf{w}_\alpha}(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ and $D_{\mathbf{w}_\alpha}(\mathbf{z}_1, \ldots, \mathbf{z}_n)$ respectively, the $D_{\mathbf{w}_\alpha}(\mathbf{y}_1 + \mathbf{z}_1, \ldots, \mathbf{y}_n + \mathbf{z}_n)$ is contained by the heptagon.

Chapter 6

A General Solution for Robust Linear Programs with Distortion Risk Constraints

In this chapter, we are also investigating linear optimization problems that have random parameters, however, in a general case, where they include $m \geq 1$ constraints. In constructing a robust solution $\mathbf{x} \in \mathbb{R}^d$, we control the risk arising from violations of the constraints. This risk is measured by set-valued risk measures, which extend the usual univariate coherent distortion (= spectral) risk measures to the multivariate case. To obtain a robust solution in d variables, the linear goal function is optimized under the restrictions holding uniformly for all parameters in a d-variate uncertainty set. This set is built from uncertainty sets of the single constraints, each of which is a weighted-mean trimmed region in $\mathbb{R}^d$ and can be efficiently calculated. Furthermore, a possible substitution of violations between different constraints is investigated by means of the admissable set of the multivariate risk measure. In the case of no substitution, we give an exact geometric algorithm, which possesses a worst-case polynomial complexity. We extend the algorithm to the general substitutability case, that is, to robust polyhedral optimization. Similarly to the single-constraint algorithm from the previous chapter, the consistency of the approach is shown for generally distributed parameters. Finally, an application of the model to supervised machine learning is discussed. The chapter is mostly based on Bazovkin and Mosler (2015).

6.1 Motivation

6.1.1 The robust model

In the last decade, much progress has been made in the field of *robust linear optimization*, that is, in finding worst-case solutions under uncertain side conditions. A wide spectrum of models and methods has been proposed. Recent developments in theory and practice are reviewed by Gabrel et al. (2014). For a systematized collection of most significant ones, see Bertsimas et al. (2011). An important part of the literature uses *risk measures* to quantify the uncertainty of violations of side conditions; see, e.g., Mosler and Bazovkin (2014), Bertsimas and Brown (2009), and Natarajan et al. (2009). Such risk measures are flexible and allow an immediate interpretation. They can be properly selected and tuned to control the relevant sources of uncertainty. Then, essentially, the goal function is optimized under the restriction that the risk of violation stays within acceptable bounds.

The present chapter contributes to this strand. We generalize the approach of Mosler and Bazovkin (2014), which is described in Chapter 5, where a single-constraint optimization was solved, to much more general restrictions. In doing this, for each linear restriction of a given linear program a so-called *uncertainty set* of parameters is constructed, which consists of all possible values of the unknown coefficients that are acceptable for the specified risk level of constraint violation. We employ *multivariate coherent distortion risk* measures, which are extensions of the usual coherent distortion risk measures or, equivalently, coherent spectral risk measures. The uncertainty sets regarding these measures are convex bodies and come out to coincide with *weighted-mean (WM) trimmed regions*. WM regions, as recently developed by Dyckerhoff and Mosler (2011), describe a multivariate distribution by regions of different *depth* (= centrality). They can be exactly calculated in any dimension (Bazovkin and Mosler (2012a)).

Various other notions of uncertainty sets have been proposed in the recent literature; a review in the context of portfolio optimization is given in Fabozzi et al. (2010). These approaches define the uncertainty set similar to a confidence set, describing the uncertainty by special functionals (e.g. φ-divergences in Ben-Tal et al. (2013)) or the uncertainty in parameters of some parametric distributional assumptions (e.g. Delage and Ye (2010)). In our approach we avoid such assumptions and represent the uncertainty of constraint violations in a purely nonparametric way, *viz.* by depth-based central regions. To obtain a numerical solution of a given optimization problem we employ the well developed geometric machinery of central regions calculation.

Originally, we are given the following stochastic linear program:

$$\mathbf{c}'\mathbf{x} \longrightarrow \min \quad s.t.\ \tilde{\mathbf{A}}\mathbf{x} \geq \tilde{\mathbf{b}}\,, \tag{6.1}$$

assuming that $\tilde{\mathbf{A}}$ is an $m \times d$ matrix having stochastic entries and that $\tilde{\mathbf{b}} \in \mathbb{R}^m$ may be stochastic, too. As earlier, random variables are marked with a tilde: For example, $\tilde{\mathbf{b}}$ will denote the stochastic right-hand side vector, while $\mathbf{b}$ is used for a deterministic value of it.

To simplify the readability of this chapter, we recall some notions from the chapter 5. To obtain a risk-constrained stochastic linear program, we use one of the following forms of constraints:

$$\rho^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) \leq \mathbf{0}\,, \quad \text{resp.} \tag{6.2}$$

$$\rho^1(\tilde{\mathbf{A}}_j\mathbf{x} - b_j) \leq 0\,, \quad j = 1 \ldots m\,, \tag{6.3}$$

where $\tilde{\mathbf{A}}_j$ denotes the j-th row of $\tilde{\mathbf{A}}$, and ρ^m is a risk measure taking values in $\mathbb{R}^m$, $m \geq 1$. An SLP that minimizes $\mathbf{c}'\mathbf{x}$ subject to the restrictions (6.2) or (6.3) is called a *risk-constrained stochastic linear program.*

A crucial point is the choice of the risk measure. As an alternative to it, we may control the probability of satisfying all restrictions by a *joint chance constraint*,

$$\text{Prob}[\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b} \geq \mathbf{0}] \geq 1 - \alpha\,. \tag{6.4}$$

Limiting the violation probability by some fixed α we obtain the chance-constrained linear program. The latter program allows in general no easy solution, as the possible stochastic dependency between coefficients of different linear constraints (i.e., between the rows of $\tilde{\mathbf{A}}$) complicates the problem. To boost the tractability, we may neglect such dependencies and split (6.4) into *separate chance constraints*,

$$\text{Prob}[\tilde{\mathbf{A}}_j\mathbf{x} - b_j \geq 0] \geq 1 - \alpha_j\,, \quad j = 1 \ldots m\,. \tag{6.5}$$

Here, we could say that each chance constraint limits the risk regarding a single side condition by imposing a maximum probability α_j on its violation. However, it is known that even in this situation the problem turns out to be computationally tractable only for special distributions of parameters. That is why we leave such models aside from consideration, staying with well-suitable risk measures.

In the current chapter, we consider no single risk measure but a whole class of such measures, the *coherent distortion risk measures* (see Definition 5.8). Given a random variable Y that has an empirical distribution on the ordered values $y_{(1)}, y_{(2)}, \ldots, y_{(n)}$, this definition of a distortion risk measure ρ becomes

$$\rho(Y) = -\sum_{i=1}^{n} w_i \, y_{(i)} \,, \tag{6.6}$$

with weights $w_i = r(y_{(i)}) - r(y_{(i-1)})$. Coherent distortion risk measures possess certain desired properties: monotonicity, translation invariance, law invariance, positive homogeneity and subadditivity. The last two properties imply a most prominent postulate of risk measurement: coherence, that is, the risk measure decreases with diversification. In a more general context, the risk measure used here can be seen as a *quality measure* (cf. Kall and Mayer (2010)). Our choice of the quality measure, besides its generality, possesses a clear interpretation and always generates a convex program. Later we will demonstrate that our approach in fact suites an even more general robust program that not only copes with linear stochastic restrictions, but also those of a *robust polyhedral* type, which include robust conic restrictions as a special case.

In applications $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ usually have to be estimated from data. Here we assume that a sample of coefficient matrices $\mathbf{A}^1, \ldots, \mathbf{A}^n \in \mathbb{R}^{m \times d}$ has been observed and the solution of the SLP is based on this data. The data is mentioned as an *empirical distribution* giving equal mass $\frac{1}{n}$ to $\mathbf{A}^1, \ldots, \mathbf{A}^n$, and with this data the SLP is named an *empirical risk-constrained SLP*. Similarly, when also the right hand side is stochastic, a joint sample of $(\tilde{\mathbf{A}}, \tilde{\mathbf{b}})$ is considered.

Theorem 5.2 states that the class of univariate restrictions (6.3) involving *coherent distortion risk measures* corresponds to *weighted-mean* trimmed regions in $\mathbb{R}^d$ as uncertainty sets. Calculating $\mathcal{U}$ turns out to be computationally feasible due to the direct connection between the uncertainty set and a trimmed region, which can be efficiently determined by the algorithm of Bazovkin and Mosler (2012a), which is described in the chapter 2.

The risk measure ρ defines the vector $\mathbf{w}_\alpha$ uniquely and, by this, the trimmed region $D_{\mathbf{w}_\alpha}$. The further steps of the algorithm of Mosler and Bazovkin (2014) are essentially standing on the search for an intersection of a ray with the uncertainty set $\mathcal{U} = D_{\mathbf{w}_\alpha}$. The solution is characterized by the normal to the facet of $\mathcal{U}$ that is intersected by the line in direction of the vector $\mathbf{c}$.

The goal of this chapter is to develop a similar approach to the general SLP (6.2). Specifically, we

1. *generalize* the analysis of the single-constraint SLP to multiple risk constraints ($m \geq 2$);
2. *construct* a geometric algorithm to solve the multi-constraint problem (if constraints cannot be compensated by each other, i.e. in the unsubstitutability case, the algorithm operates in the same dimension d as the single-constraint procedure does);
3. *extend* the robust multi-constraint linear optimization to *robust polyhedral optimization* (which covers the substitutability case);
4. *estimate* the uncertainty set and the robust solution consistently.

The further material is organized as follows. The construction of the solution for the multi-constraint SLP is described in Section 6.2. The formal algorithm is given in the subsequent Section 6.3. The same section reviews the extension of the algorithm to a program with stochastic right-hand side. Its consistency with the generally distributed data is proven in Subsection 6.3.5. Finally, Section 6.4 is devoted to a discussion of the algorithm and an application to supervised learning.

6.2 Multiple constraints

6.2.1 A general model

Consider the SLP (6.1) with $m \geq 2$ constraints and deterministic right-hand side $\mathbf{b}$. We aim at generalizing Theorem 5.2 and eliminating uncertainty by a robust linear program as follows:

$$\mathbf{c}'\mathbf{x} \longrightarrow \min \quad s.t.\ \mathbf{A}\mathbf{x} \geq \mathbf{b} \ \text{ for all } \mathbf{A} : \delta(\mathbf{A}) \in \mathcal{U}\,, \tag{6.7}$$

where $\delta(\mathbf{A}) = (\mathbf{A}_1, ..., \mathbf{A}_m)' \in \mathbb{R}^{m \cdot d}$ is the vectorized matrix $\mathbf{A}$.

Again, a proper uncertainty set $\mathcal{U}$ has to be constructed. In doing this, we first specify the m-variate risk measure, which is set-valued. Cascos and Molchanov (2007) have shown that certain multivariate risk measures correspond to trimmed regions of the considered distribution. In particular, the m-variate set-valued analogue μ^m of a coherent spectral risk measure is defined through a weighted-mean (WM) region $D_{\mathbf{w}_\alpha}$ in the following manner:

$$\mu^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) = -\Big(D_{\mathbf{w}_\alpha}(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) \oplus \mathbb{R}^m_+\Big) \subset \mathbb{R}^m. \tag{6.8}$$

It can be interpreted as the set of deterministic vectors in $\mathbb{R}^m$ which, being added to the random variable $\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}$, cannot shift its α-region out of the negative orthant $\mathbb{R}^d_-$. In other words, which cannot, to the given precision level, guarantee to avoid a strictly negative outcome of the shifted random variable. It is easy to see, that in dimension one $\mu^1(\tilde{\mathbf{a}}'\mathbf{x} - b)$ is the half-line bounded above by $\rho^1(\tilde{\mathbf{a}}'\mathbf{x} - b)$, where ρ^1 is a coherent spectral measure of a univariate risk. Thus, (6.8) can be regarded as a set-valued extension of a univariate distortion risk measure to multiple dimensions[1]. This measure has been considered in detail in Chapter 4. In this chapter, we use μ for denoting set-valued, and ρ for denoting vector-valued or real-valued measures.

If a linear program (6.1) has more than one stochastic constraints, we must consider not only that some or all of them may be violated, but also that the degree of violation of a restriction may be offset against that of another restriction. That is, the decision maker, in evaluating a possible solution, may compensate the missing strictness of one constraint by the fact that another constraint or a group of constraints is more strictly satisfied. In this, the *values* of single constraint satisfaction are regarded as *substitutable* by the decision maker, and his or her task in selecting a solution includes some kind of diversification regarding the constraints. Note that this possible value compensation between the constraints has nothing to do with a potential stochastic dependency among the parameters of different constraints.

To include the possibility of value substitution in our model, we introduce a multivariate utility (= negative loss) function $u : \mathbb{R}^m \to \mathbb{R}$ that evaluates the violations $v_1, \ldots, v_m$ of the m constraints. Consider $\mathcal{F} = \{\mathbf{v} : u(\mathbf{v}) \geq 0\}$ as the set of admissable violations. If u is a quasiconcave function, $\mathcal{F}$ is convex. Later we will specialize $\mathcal{F}$ to be a convex polyhedron, see (6.10). The marginals may or may not substitute each other. This fact actually affects the form of $\mathcal{F}$.

Using $\mathcal{F}$, we rewrite the joint risk constraint (6.2) as follows[2]:

$$-\mu^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) \subset \mathcal{F}\,. \tag{6.9}$$

[1] Rüschendorf (2013) proposes a different notion of a multivariate distortion risk measure, which is scalar-valued: Given a d-variate distribution having p.d.f. F, he considers the level set $Q(t)$ of F at level t and defines some scalar measure of $Q(t)$ as the t-quantile. Then, based on these scalar-valued quantiles, he introduces multivariate risk measures in the same way as univariate ones.

[2] Here, $\mathcal{F}$ coincides with the admissable set of the vector-valued multivariate risk measure ν introduced in Bazovkin (2014) (see also Chapter 4). This measure consists in the smallest vector which, when being added to the random returns vector $\mathbf{Y}$, puts $-\mu^m(\mathbf{Y} + \nu(\mathbf{Y}))$ into the admissable set. The latter is the set of returns which appear to be acceptable to the risk taker (or the regulator). In this model, we obtain (6.9) with only μ^m and $\mathcal{F}$ being involved:

$$\nu(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) \leq \mathbf{0} \quad \Longleftrightarrow \quad -\mu^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) \subset \mathcal{F}\,.$$

In fact, the set $\mu^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b})$ is contained in the set of all violation vectors $\mathbf{v} \in \mathbb{R}^m$ that are admitted. If substitution between constraints is possible, the level of substitutability may vary from full substitutability to *unsubstitutability*. It is easy to show, that the first extremal case leads to an equivalent single-constraint SLP.

Full substitutability means that u is additive, $u(\mathbf{v}) = \sum_j u_j(v_j)$, and marginal utilities u_j are linear. In this case we obtain

$$\mathcal{F}_{\text{sub}} = \{\mathbf{v} : u(\mathbf{v}) = \sum_j u_j(v_j) = \sum_j k_j \cdot v_j = \mathbf{k}'\mathbf{v} \geq 0\}$$

with some $\mathbf{k} \geq \mathbf{0}$. This reduces to a problem with a single constraint $(\mathbf{k}'\tilde{\mathbf{A}})\mathbf{x} \geq \mathbf{k}'\mathbf{b}$. Here the admissable set $\mathcal{F}_{\text{sub}}$ is a halfspace bordered by the hyperplane passing the origin and having normal $\mathbf{k}$. In the second extreme case (unsubstitutability) the admissable set is the positive orthant, $\mathcal{F}_{\text{unsub}} = \mathbb{R}^m_+$. Solving the SLP in this case will be the subject of Section 6.3. In the intermediate case of the partial substitutability $\mathcal{F}$ is a set lying in $\mathcal{F}_{\text{sub}}$ and containing $\mathcal{F}_{\text{unsub}}$.

To sum up, for obtaining a general solution of the multi-constraint SLP we have to consider different levels of substitutability among the violation of constraints. It turns out that the general substitutability case can be reduced to unsubstitutability via a special transformation of the model. In the next subsection we will define the transformation and demonstrate this fact. After that, to manage the complete task, we solve the SLP with unsubstitutability.

6.2.2 The general substitutability case

Now we consider the case that violations of constraints can be balanced against each other. Let us assume that the set $\mathcal{F}$ of feasible violations is a convex polyhedron, characterized by linear inequalities,

$$\mathcal{F} = \{\mathbf{y} \in \mathbb{R}^m : \mathbf{p}_k'\mathbf{y} \geq d_k, \quad k = 1 \dots K\} \tag{6.10}$$

with some $\mathbf{p}_1, \dots, \mathbf{p}_K \in \mathbb{R}^m_+$ and $d_1, \dots, d_K \in \mathbb{R}$, that is, $\mathcal{F}$ is an *upper convex polytope*.

Proposition 6.1. *Let $\rho^1(Z)$ denote the upper border of the halfline $\mu^1(Z) \subset \mathbb{R}$. Then it holds: $-\mu^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b}) \subset \mathcal{F}$ if and only if*

$$\rho^1(\mathbf{p}_k'\tilde{\mathbf{A}}\mathbf{x} - \mathbf{p}_k'\mathbf{b}) \leq -d_k\,, \quad \textit{for all } k = 1 \dots K\,. \tag{6.11}$$

Proof. Let h_S^m denote the support function of a set S in $\mathbb{R}^m$. For any $\mathbf{y} \in -\mu^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b})$ and $k = 1 \dots K$ we obtain:

$$\begin{aligned}
\mathbf{p}_k'\mathbf{y} &\geq \min_{\mathbf{z} \in -\mu^m(\tilde{\mathbf{A}}\mathbf{x}-\mathbf{b})} \{\mathbf{p}_k' \cdot \mathbf{z}\} \\
&= h^m_{-\mu^m(\tilde{\mathbf{A}}\mathbf{x}-\mathbf{b})}(-\mathbf{p}_k) \\
&= h^1_{\mathbf{p}_k' \cdot \mu^m(\tilde{\mathbf{A}}\mathbf{x}-\mathbf{b})}(1) = h^1_{\mu^1(\mathbf{p}_k'\tilde{\mathbf{A}}\mathbf{x} - \mathbf{p}_k'\mathbf{b})}(1) \\
&= -\rho^1(\mathbf{p}_k'\tilde{\mathbf{A}}\mathbf{x} - \mathbf{p}_k'\mathbf{b})\,.
\end{aligned}$$

Consequently, we have $\mathbf{y} \in \mathcal{F}$ if $-\rho^1(\mathbf{p}_k'\tilde{\mathbf{A}}\mathbf{x} - \mathbf{p}_k'\mathbf{b}) \geq d_k$. Hence the "if" part of the proposition is proved. On the other hand, there exists some $\mathbf{y} \in -\mu^m(\tilde{\mathbf{A}}\mathbf{x} - \mathbf{b})$ so that the first-line inequality is met with equality. Hence the "only if" part holds, too. □

Proposition 6.1 leads to the following theorem, which provides for every general model an equivalent unsubstitutability model:

Theorem 6.2. *The SLP with risk constraint* (6.9)*, where $\mathcal{F}$ is defined by (6.10), is equivalent to an SLP with unsubstitutable constraints*

$$p_k'\tilde{\mathbf{A}} \geq p_k'\mathbf{b} \quad \textit{for } k = 1 \dots K\,. \tag{6.12}$$

Proof. According to Proposition 6.1, the SLP with joint risk constraint (6.9) is equivalent to the SLP with constraints (6.11). On the other hand, an SLP with constraints (6.12) produces the same risk constraints. □

Theorem 6.2 enables us to determine a generalized uncertainty set in the multi-constraint case.

Corollary 6.3. *The uncertainty set $\mathcal{U}$ of the matrix $\tilde{\mathbf{A}}$ in an SLP with risk constraint* (6.9) *and violations admissable set $\mathcal{F}$* (6.10) *equals*

$$\mathcal{U} = \left\{ \mathbf{A} : \delta([p_k'\tilde{\mathbf{A}}]_{k=1\dots K}) \in \mathsf{X}_{j=1..K} D_{\mathbf{w}_{\alpha_j}}(p_k'\tilde{\mathbf{A}}) \right\}. \tag{6.13}$$

Proof. Here X denotes the K-fold Cartesian product of WM regions. Due to Theorem 6.2, we can transform the general model into a model with unsubstitutability. In this case single uncertainty sets will not affect each other. In turn, each single uncertainty set is calculated as for the single-constraint SLP. □

6.2.3 The equivalent unsubstitutability case

We see that $\mathcal{U}$ splits up into K parts, which can be calculated individually. It leads to the following key representation theorem:

Theorem 6.4. (Bazovkin and Mosler, 2015) *The SLP* (6.1) *with violations admissable set* $\mathcal{F}$ *and joint risk constraint* (6.9) *is equivalent to the following problem:*

$$\mathbf{c}'\mathbf{x} \longrightarrow \min \quad s.t.\ \mathbf{A}\mathbf{x} \geq \mathbf{b} \ \textit{ for all } \mathbf{A} \in \mathcal{U}\,, \tag{6.14}$$

where $\mathcal{U}$ *is defined as in Corollary 6.3.*

Proof. Follows from Theorem 6.2, Corollary 6.3 and Theorem 5.2. □

Note that if all d_k equal zero, we get the general risk-constrained *robust conic program*. Besides this, unsubstitutability, obviously, does not imply stochastic independence of the constraints.

At this point it is reasonable to compare the present framework with that of chance-constrained problems. Such SLPs (6.5) with individual constraints are extremely critical to distributional assumptions: in most cases the program turns out to be non-convex (see, e.g., Kall and Mayer (2010)) and computationally intractable. Plausible results are recently obtained only for the elliptically distributed random coefficients. If we have a joint chance-constraint, the difficulty increases. A straight-forward approach, which distributes the common violation probability equally among the individual constraints, tends to give poor results, especially if the constraints are stochastically dependent. This example of approximative solution based on the Bonferroni inequality has been improved by Chen and Sim (2009) and Chen et al. (2010), who propose more efficient bounds for the individual probabilities of violation of constraints using results from the order statistics. But altogether these approximative methods still lack strong interpretation and universality. In contrast, our approach leads to a natural decomposition of (6.8), while remaining jointly constrained. Any stochastic dependency among parameters (and, thus, constraints) is completely feasible.

A powerful flexibilization of our model consists in varying the α's in (6.13). This is practical, since some constraints can be more tolerant to violations while others, on the contrary, are rather strict or even exact (e.g., a non-negativity requirement).

6.3 Unsubstitutable violations risks: The optimal solution

6.3.1 Generalizing the single-constraint approach

By Theorem 6.2 the general risk-constrained SLP is reduced to an SLP with unsubstitutable constraints. Therefore, it suffices to construct an algorithm for solving the latter SLP. Moreover, Theorem 6.4 reformulates the stochastic constraints in terms of an uncertainty set.

In this subsection we pursue the following idea: We want to reformulate our problem in a way that makes it similar to the single-constraint case. In doing so, we first define a *convolution set* that will play the same role as the uncertainty set in the single-constraint SLP algorithm. Then, we show how to construct an equivalent to the optimization line. Having proved the equivalence of the elements, we are able to formulate the generalization of the algorithm to the multiconstraint case.

Let us write $\mathcal{X}_i$ for the feasible set generated by the constraint i, $i = 1 \dots K$, and $\mathcal{X}$ for the common feasible set,

$$\mathcal{X} = \bigcap_{i=1}^{K} \mathcal{X}_i. \tag{6.15}$$

We aim at solving the SLP by a geometric procedure in the parameter space. For each constraint, the parameter space has dimension d, while the uncertainty set $\mathcal{U}$ lives in $\mathbb{R}^{d \cdot K}$. We will construct a set $\mathcal{G} \subset \mathbb{R}^d$ such that $\mathcal{X}$ can be rewritten as follows:

$$\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d : \mathbf{A}\mathbf{x} \geq \mathbf{b} \text{ whenever } \delta(\mathbf{A}) \in \mathcal{U}\} = \{\mathbf{x} \in \mathbb{R}^d : \mathbf{a}'\mathbf{x} \geq 1 \; \forall \mathbf{a} \in \mathcal{G}\}. \tag{6.16}$$

$\mathcal{G}$ is obtained by convolving the general uncertainty set $\mathcal{U}$ into $\mathbb{R}^d$; we will call $\mathcal{G}$ the parameter *convolution set*. $\mathcal{U}$ is then decomposed into the sets $\mathcal{U}_i, i = 1, \dots, K$, that can be separately calculated. This dimension reducing construction is possible as the constraints are not substitutable.

The proper way here is to represent $\mathcal{G}$ as an image of $\mathcal{X}$ in the parameter space. According to (6.15), all $\mathcal{X}_i$ are combined in one space. Combining $\mathcal{U}_i$ in the parameter space becomes possible if any parameter $\mathbf{a}$ contained by other uncertainty sets corresponds to the same $X_{\mathbf{a}} = \{\mathbf{x} : \mathbf{a}'\mathbf{x} \geq b\}$ in each case. This condition holds if the right-hand sides b_i are the same for all constraints. If $\mathbf{b} > \mathbf{0}$, we multiply all sets $\mathcal{U}_i$ with $\frac{1}{b_i}$ and obtain $b = 1$, without changing the set of feasible solutions. If $\mathbf{b} \not> \mathbf{0}$ we apply the transform (6.19), which is described in the next subsection.

$\mathcal{G}$ contains the union of $\frac{1}{b_i}\mathcal{U}_i, i = 1 \dots K$. Thus, according to (6.15) and similar to Lemma 1 in Mosler and Bazovkin (2014):

$$\mathcal{X} = \bigcap_{\mathbf{a}\in\mathcal{G}} X_{\mathbf{a}} = \bigcap_{\mathbf{a}\in \operatorname{ext}\mathcal{G}} X_{\mathbf{a}},$$

where $\operatorname{ext}\mathcal{G}$ denotes the set of *extreme points* of $\mathcal{G}$, which for a convex body corresponds to its set of vertices.

This proves that $\mathcal{G}$ has similar properties as the uncertainty set of the single-constraint SLP. In particular, any convex combination of two points in $\mathcal{G}$ belongs to $\mathcal{G}$. We obtain:

$$\mathcal{G} = \operatorname{conv}\left\{\bigcup_{i=1}^{K} \frac{1}{b_i}\cdot\mathcal{U}_i\right\}. \tag{6.17}$$

This transformation of $\mathcal{X}$ is familiar to polar duality (see, e.g., Grünbaum (2003)). In our robust problem we need no explicit representation of $\mathcal{X}$ and profit from the ready machinery of WM regions that allows the direct construction of $\mathcal{G}$ in the parameter space.

The next step is constructing the optimization line, which is equivalent to the single-constraint case and is the line passing through the origin in direction $\mathbf{c}$. Actually, φ is the locus of points that are dual to hyperplanes with the normal $\mathbf{c}$. However, combining the $\mathcal{U}_i$ in one parameter space requires individual transforms of each constraint's parameter spaces, thus resulting in different φ_i. Observe that

- $\varphi_i \equiv s\cdot\varphi_i$ for any $s \neq 0$,
- $\varphi_i \equiv \varphi$ for all i.

Hence all lines φ coincide. Further, they are invariant to the affine transform in (6.17). Therefore, the search of the optimum on $\mathcal{G}$ equals the search on $\mathcal{U}$ in the single-constraint SLP.

In concluding this subsection, we turn to the deterministic case, where each $\mathcal{U}_i$ degenerates to a one-point-set $\{\mathbf{a}_i\}$.[3] The steps of our procedure stay the same but allow some simplifications. For example, calculating $\mathcal{G}$ according to (6.17) reduces to the simple *quickhull* routine.[4]

In many practical tasks one has to consider some additional deterministic constraints, in particular, those of nonnegativity type. In such a setting, a group of trivial constraints

[3] Consequently we do not need to calculate WM regions (see Figure 6.1).

[4] Thus, the above approach can be also employed as an alternative to the regular simplex method.

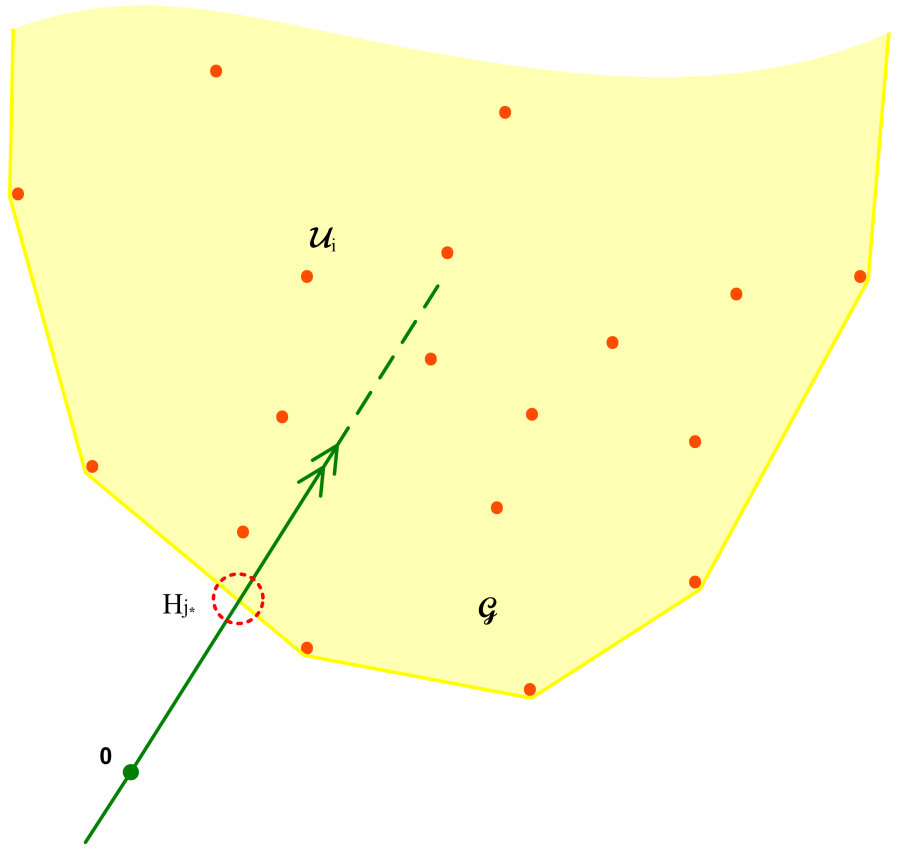

FIGURE 6.1: Alternative to the simplex algorithm.

$x_k \geq 0, k \in \mathcal{J}$, is mapped to a finite cloud of points in the parameter space without calculating the WM region. The latter implies that such constraints do not significantly influence the algorithm's computational complexity.

6.3.2 Relaxing the right-hand side

In this section we show that the model (6.1) of the SLP also covers the case of a random coefficient vector $\mathbf{b}$, denoted by $\tilde{\mathbf{b}}$, as the restriction $\tilde{\mathbf{A}}\mathbf{x} \geq \tilde{\mathbf{b}}$ is equivalent to

$$\left[\tilde{\mathbf{A}} \quad \mathbf{1} - \tilde{\mathbf{b}}\right] \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \geq \mathbf{1}\,. \tag{6.18}$$

By this we obtain an SLP with deterministic right-hand side equal to $\mathbf{1}$. Now the solution vector has $d+1$ components, the last of which is fixed to 1. Geometrically the solution corresponds to the intersection of the feasible set with the d-dimensional hyperplane $\{\mathbf{x} : x_{d+1} = 1\}$, that is, the task is actually solved on a convex polytope of affine dimension d.

Formally speaking, we have to solve an SLP of the form (6.1) with an additional equality constraint. For this we propose a concise geometrical approach for solving such a problem. The idea is to modify the vector $\mathbf{c}$ so that it gives us the solution of the conditional task.

First, we replace $\mathbf{c}$ with a vector $\bar{\mathbf{c}}$ equal to the normal of the facet that lies on the hyperplane $\{(\mathbf{x}', 1)' \in \mathbb{R}^{d+1}\}$. By this, we get a non-unique solution, which also contains the sought-for optimum $\mathbf{x}^*$. Looking simultaneously to the parameter space, we see an intersection of φ with $\mathcal{G}$ (will be defined in the next section) at the point $(0, \dots 0, 1)'$. To get rid of this non-uniqueness, we perform a small rotation of $\bar{\mathbf{c}}$ towards $\mathbf{x}^*$. The latter amounts to rotating $\bar{\mathbf{c}}$ towards $(\mathbf{c}', 0)'$, that is, to some new vector $\mathbf{c}_\epsilon = (1-\epsilon)\bar{\mathbf{c}} + \epsilon \cdot (\mathbf{c}', 0)'$ with some $0 < \epsilon < 1$.

Proposition 6.5. *Let $\tilde{\mathbf{b}}$ be a stochastic vector of dimension K. Then (6.1) is equivalent to the following model: There exists $\bar{\epsilon} > 0$ such that*

$$\mathbf{c}_\epsilon' \begin{bmatrix} \mathbf{x} \\ x_{d+1} \end{bmatrix} \longrightarrow \min \tag{6.19a}$$

$$s.t. \quad \begin{bmatrix} \tilde{\mathbf{A}} & \mathbf{1} - \tilde{\mathbf{b}} \\ \mathbf{0}' & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ x_{d+1} \end{bmatrix} \geq \mathbf{1}\,, \tag{6.19b}$$

where $\mathbf{c}_\epsilon = (1-\epsilon)(0, \dots 0, 1)' + \epsilon \cdot (\mathbf{c}', 0)'$, $0 < \epsilon < \bar{\epsilon}$.

Proof. The constraints of (6.1) are equivalent to (6.18). Obviously, if $x_{d+1} = 1$ holds, (6.19b) is also equivalent to (6.18). To show that there exists such a small $\epsilon > 0$ that the last inequality in (6.19b) necessarily turns into an equality and, as a consequence, (6.19b) transforms into (6.18), we first set ϵ to zero. The corresponding program has a trivial non-unique solution, because the vector $\mathbf{c}_\epsilon$ equals $\bar{\mathbf{c}}$, which was shown above to pick a whole facet of the feasible set for a solution.

Some small rotation of $\mathbf{c}_\epsilon$ shifts the solution to the border of this facet, however remaining on it. This rotation is given by setting $\epsilon > 0$, i.e. combining the initial vector with the augmented vector $\mathbf{c}$, namely $(\mathbf{c}', 0)'$. The both facts together guarantee fixing x_{d+1} to 1 while optimizing (6.19) with $\mathbf{c}_\epsilon$. In this situation, (6.19b) reduces to (6.18). Moreover, the objective function turns into $\epsilon \mathbf{c}'\mathbf{x} + 1 - \epsilon$, which is, up to a constant, equivalent to the objective of (6.1). This proves the equivalence of (6.19) and (6.1). □

Let us now imagine the procedure shown above in the parameter space (see Figure 6.2). The additional $\mathcal{U}_{K+1}$ is just a point $(0, \dots 0, 1)'$. The virtual optimization vector $\mathbf{c}_\epsilon$ generates a line $\varphi_\epsilon = \epsilon \cdot \varphi + (1-\epsilon) \cdot \{\mathbf{x} : x_j = 0, j = 1 \dots d\}$, that is, an equivalent

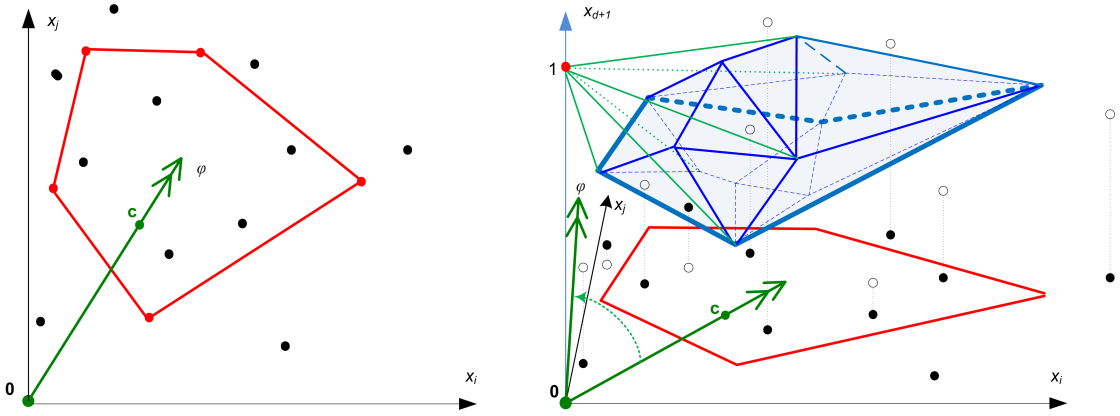

FIGURE 6.2: Adding a dimension.

affine combination of φ and an axis passing through the point $(0,\dots 0,1)'$. φ_ϵ intersects necessarily the cone (part of the surface of $\mathcal{G}$) having $(0,\dots 0,1)'$ as its apex. The respective facet of $\mathcal{G}$ determines the optimum $\mathbf{x}^*$ similar to step D.c. of the algorithm in Subsection 6.3.3 below.

6.3.3 The algorithm

Prerequisites:

In solving the *general robust polyhedral optimization* problem with an arbitrary $\mathcal{F}$, we first modify our set of constraints and the vector $\mathbf{b}$ according to Theorem 6.2, thus obtaining the $K \times d$ matrix $\tilde{\mathbf{A}}$ and the K-dimensional vector $\mathbf{b}$.

Without loss of generality, the algorithm is applied to a minimization problem with all constraints of the same type "$\geq$". If either $\mathbf{b} \notin \mathbb{R}^K_+$ or $\mathbf{b}$ is stochastic, the pretransformation of Proposition 6.5 should be applied first. In the sequel both modifications (including that for stochastic right-hand side $\tilde{\mathbf{b}}$) are assumed to be done if necessary. Hence, we have to solve an SLP of form (6.19),

$$\mathbf{c}'_\epsilon \begin{bmatrix} \mathbf{x} \\ x_{d+1} \end{bmatrix} \longrightarrow \min$$
$$s.t. \quad \begin{bmatrix} \tilde{\mathbf{A}} & \mathbf{1} - \tilde{\mathbf{b}} \\ \mathbf{0}' & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ x_{d+1} \end{bmatrix} \geq \mathbf{1}\,,$$

with $\mathbf{c}_\epsilon = (1-\epsilon)(0,\dots 0,1)' + \epsilon \cdot (\mathbf{c}',0)'$, $0 < \epsilon < \bar{\epsilon}$, where $\bar{\epsilon}$ is a small positive constant.

Input

- a vector $\mathbf{c} \in \mathbb{R}^d$ of coefficients of the goal function,

- a combined sample from $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$: $\left[\begin{smallmatrix}\mathbf{a}^{ji}\\ 1-b_j^i\end{smallmatrix}\right]_{j=1\ldots K}$, $i=1,\ldots,n$,
- a distortion *risk measure* μ^K, given by a name or an explicit weight vector.

Output

- A part of the convolution set $\mathcal{G}$ that includes the optimal solution,
- the *optimal solution* $\mathbf{x}^*$.

Steps of the Algorithm

A. Construct the individual uncertainty sets:

   a. Determine, by the algorithm of Bazovkin and Mosler (2012a), $\{\mathcal{U}_j\}_{j=1\ldots K}$, i.e. the WM regions $\{D_{\mathbf{w}_{\alpha_j}}\}_{j=1\ldots K}$ of each random vector $\left[\begin{smallmatrix}\tilde{\mathbf{a}}^j\\ 1-\tilde{b}_j\end{smallmatrix}\right]$ having an empirical distribution on $\left[\begin{smallmatrix}\mathbf{a}^{j1} & \ldots & \mathbf{a}^{jn}\\ 1-b_j^1 & \ldots & 1-b_j^n\end{smallmatrix}\right]$, $j=1\ldots K$.

   b. Add the uncertainty set for the $(K+1)$-th deterministic constraint in (6.19b): $\mathcal{U}_{K+1}=\{(\mathbf{0}',1)'\}$.

   c. For each nonnegativity restriction $x_j\geq 0$, add d point-sets $\mathcal{N}_j=\{(\mathbf{e}_j',1)'\}$ to the uncertainty sets, where $\mathbf{e}_j$ is the j-th unit vector in $\mathbb{R}^d$.

B. Calculate the convolution set $\mathcal{G}$ represented by its facets $\{(\mathbf{n}_j;d_j)\}_{j\in G}$:

   a. Take the representation of $\{\mathcal{U}_i\}_{i=1\ldots K}$ by their vertices and put them into the same space after having rescaled them according to (6.17).

   b. Calculate the convex hull of the set using the standard *quickhull* (Barber et al., 1996) or some *divide-and-conquer* algorithm (Grünbaum, 2003).

C. Impose the optimization ordering on the space of parameters, creating the dual representation of the optimization vector $\mathbf{c}$. It is a line φ connecting the origin $(\mathbf{0}',0)'$ and the point $(\epsilon\cdot\mathbf{c}',1-\epsilon)'$, with a small $\epsilon>0$.

D. Search for a facet H_{j_*} of $\mathcal{G}$ that is intersected by φ (see Figure 6.3). Its dual defines the sought-for optimal solution $\mathbf{x}^*$:

   a. Define a set of facets $\mathcal{G}_{sel}$ to be analysed: This may be either $\mathcal{G}$ itself or its part where the intersection is expected; $\mathcal{G}_{sel}=\{(\mathbf{n}_j,d_j): j\in J_{sel}\}$.

   b. Take some $\mathbf{u}=\lambda\mathbf{c}, \lambda\geq 0$, outside the augmented $\mathcal{G}$.
   Find the $j_*=\arg\max_j\{\frac{d_j}{\mathbf{u}'\mathbf{n}_j}\}_{j\in J_{sel}}$.

   c. $\mathbf{x}^*=\frac{1}{d_{j_*}}\cdot\mathbf{n}_{j_*}$ is the optimal *solution*.

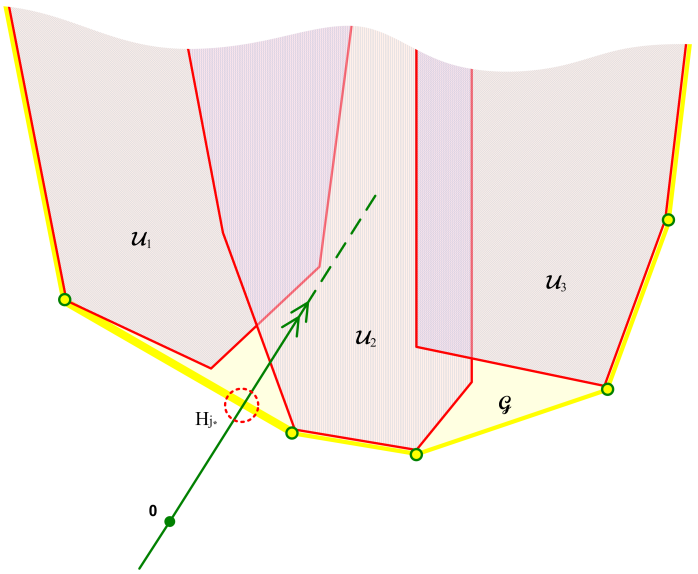

FIGURE 6.3: Obtaining the solution using the convex hull computation routine.

d. If there is no intersection, the *solution is at infinity.*

e. If $(\mathbf{0}', 1)' \in \mathcal{G}$, there is *no solution.*

For efficient calculation of the intersection of φ with $\mathcal{G}$ we can apply an approximative procedure which converges to the precise solution. The procedure finds a facet of some scaled $\mathcal{U}_i$ that is closest from outside to the sought-for facet of $\mathcal{G}$. Obviously, if the facet is part of some uncertainty set, the obtained solution is optimal.

6.3.4 Complexity

The complexity of the algorithm is firstly determined by the routine for the convex hull. Before analyzing the general algorithm, we take a look at its deterministic counterpart. In solving a deterministic LP we have to calculate the convex hull of $K+1$ points, which has a worst-case complexity of $\mathcal{O}(K \log K + K^{\lceil \frac{d}{2} \rceil})$; see Chazelle (1993). The naive linear search on the set of n facets has a complexity of $\mathcal{O}(nd)$. Consequently, the worst-case complexity of our algorithm amounts to $\mathcal{O}(d \cdot K^{\lceil \frac{d}{2} \rceil})$, which is polynomial. It is well known (e.g. Borgwardt (2001)) that no variant of the simplex method exists that solves an LP with polynomial complexity. Only ellipsoid and interior-point methods (using randomizations) can achieve polynomial complexity.

For the general algorithm is is very natural to use "divide and conquer" algorithms, which construct the convex hull of the whole data out of convex hulls of subsets of

data. Such procedures have best complexities in dimensions 2 and 3, namely $\mathcal{O}(n \log n)$. For example, the standard quickhull algorithm has complexity between $\mathcal{O}(n \log n)$ and $\mathcal{O}(n^2)$, depending on the input. However, to our knowledge, such "divide and conquer" algorithms are available in the literature only for dimensions up to 5 (see, e.g., Buckley (1988)).

In fact, we need not calculate the whole convex hull, because we are interested only in the normal to the hyperplane at the intersection (even the actual facet is not interesting for us). This is why we can substantially reduce the complexity by cutting off the region of the intersection by an ellipsoid or a cylinder having axis φ. The worst-case complexity of the general algorithm is also polynomial, however of a high power, which can cause difficulties for large-scale problems.

6.3.5 Robust SLP for generally distributed coefficients

In the previous sections we assumed the random parameters $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ to follow an empirical distribution based on observations $\{\mathbf{A}^1, \ldots, \mathbf{A}^n\}$ and $\{\mathbf{b}^1, \ldots, \mathbf{b}^n\}$. Now we want to consider an SLP (6.1), where $(\tilde{\mathbf{A}}, \tilde{\mathbf{b}})$ follows a general probability distribution P and realizations are randomly sampled from this distribution.

Actually, Mosler and Bazovkin (2014) have shown that the individual uncertainty set $\mathcal{U}_j$ is a consistent estimator of its population counterpart. Convergence is almost surely in the Hausdorff sense, which is based on the law of large numbers for weighted-mean regions (Dyckerhoff and Mosler (2012)). Here, our general algorithm constructs $\mathcal{G}_n$ as the convex union of individual uncertainty sets, which, obviously, also converges almost surely in the Hausdorff sense to $\mathcal{G}$:

Proposition 6.6. *$\mathcal{G}_n$ converges to $\mathcal{G}$ almost surely in the Hausdorff sense.*

Also the cutting point, where the line φ hits the convolution set $\mathcal{G}$, is consistently estimated by our algorithm. A potential complication lies in the fact that the surface of $\mathcal{G}$ is, in general, not smooth. That is why the optimal solution $\mathbf{x}^*$, which, obviously, is defined by the tangent hyperplane at the cutting point can be ambiguous. However, even in such situations, the algorithm automatically selects a unique facet of $\mathcal{G}$ determining $\mathbf{x}^*$.

6.4 Conclusion and application

A new geometric algorithm is proposed for robust linear optimization under distortion risk constraints. The algorithm constructs an uncertainty set in the parameter space,

which measures the risk arising from non-deterministic parameters in the original linear constraints. The randomness may affect the coefficient matrix $\mathbf{A}$ as well as the right hand side $\mathbf{b}$. In our setting a multivariate coherent distortion risk measure is applied to the joint distribution of the parameters. This results in uncertainty sets for each single constraint, which are so called weighted-mean trimmed regions. The multi-constraint uncertainty set then comes out as the convex hull of the union of rescaled single-constraint uncertainty sets. It is determined by calculating the relevant parts of weighted-mean trimmed regions, which is done by the algorithm of Bazovkin and Mosler (2012a). (Note that the uncertainty set needs not be determined from an external sample; alternatively it can be introduced explicitly by the optimizer. In this case the algorithm starts with step C.)

The algorithm can be applied to multi-constraint as well as single-constraint problems. Also, as a special case, deterministic linear optimization problems are solved by the algorithm. To cope with substitution in evaluating the violation of different constraints, a variant of the model is introduced, which is mentioned as *robust polyhedral optimization*.

We conclude the investigation with an efficient application of our optimization model and algorithm to classification problems. Our procedure can be applied to supervised machine learning as a robust alternative to the support vector machine (SVM). The basic problem is: Two classes of points are given in the Euclidean d-space $Q_1 = \{x_1, \ldots, x_{n_1}\}$ and $Q_2 = \{y_1, \ldots, y_{n_2}\}$. A rule has to be constructed by which any new point x is classified to one of those classes Q_1 and Q_2. The classical SVM of Vapnik (1998) determines a hyperplane that discriminates the two classes linearly in a higher-dimensional space and serves as a separator for classifying new points. Technically, this approach results in a convex *quadratic program*. To tackle the problem in a robust way, mostly methods of replacing each point by its neighbourhood are proposed in the literature; see e.g. Ben-Tal et al. (2009). In contrast, we consider no single points of the training classes as uncertain, but the whole classes, and observe the points as a sample of the random variables defining the classes. Besides having a better interpretation, we obviously can expect getting less constraints. In fact, it turns out that the robust SVM can be represented as a robust linear program with two risk constraints. To achieve these representation, we start with the following linear program:

$$\begin{aligned} & C \to \max_{\omega, C} \\ s.t. & \begin{cases} \omega' \mathbf{x}_i + b \geq C, & \mathbf{x}_i \in Q_1, \\ \omega' \mathbf{y}_j + b \leq -C, & \mathbf{y}_j \in Q_2. \end{cases} \end{aligned} \tag{SVM1}$$

Generally, the solution of (SVM1) does not coincide with the solution of the usual *quadratic* program. Like Vapnik's SVM, (SVM1) produces a central separating hyperplane that lies between the classes. However, it does not necessarily minimize the Euclidean distances between support vectors, as the Euclidean distance is not the only possible criterion here. Note that the Euclidean distance is not easily interpreted when the data have been transformed into a higher-dimensional feature space, as it is done by SVM.

To control the quality of our proposed solution, we rewrite the model (SVM1) as a stochastic linear program:

$$\begin{aligned} \mathbf{0}' \cdot \mathbf{y} - 1 \cdot z &\to \min_{\mathbf{y},z} \\ s.t. \begin{cases} \tilde{\mathbf{a}}_1' \cdot \mathbf{y} - 1 \cdot z \geq & -1, \\ (-\tilde{\mathbf{a}}_2)' \cdot \mathbf{y} - 1 \cdot z \geq & 1\,, \end{cases} \\ \text{where} \quad \tilde{\mathbf{a}}_1 \sim Q_1,\ \tilde{\mathbf{a}}_2 \sim Q_2. \end{aligned} \tag{SVM2}$$

$\tilde{\mathbf{a}} \sim Q$ means that $\tilde{\mathbf{a}}$ has an empirical distribution on the finite set Q. Following our approach, we next remove the negative values in $\mathbf{b} = (-1\ 1)'$. After applying the transformation of Proposition 6.5, we get:

$$\begin{aligned} [\mathbf{0}' \quad \epsilon \quad 1-\epsilon]\ (\mathbf{y}' \quad z \quad y_{d+1})' &\to \min_{\mathbf{y},z,y_{d+1}} \\ s.t. \begin{bmatrix} \tilde{\mathbf{a}}_1' & -1 & 2 \\ -\tilde{\mathbf{a}}_2' & -1 & 0 \\ \mathbf{0}' & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} \mathbf{y} \\ z \\ y_{d+1} \end{pmatrix} &\geq \mathbf{1}\,, \\ \text{where} \quad \tilde{\mathbf{a}}_1 \sim Q_1,\ \tilde{\mathbf{a}}_2 \sim Q_2. \end{aligned} \tag{SVM3}$$

The origin $\mathbf{0}$ must not be situated between the classes, otherwise we may obtain an infinite solution. We extend this application as follows: To control the width of the margin we make $\mathbf{b}$ stochastic (instead of fixing it at $(-1\ 1)'$). The more uncertain $\mathbf{b}$, the wider is the margin of the separating hyperplane.

A soft margin is introduced as usual; see Vapnik (1998). However, in contrast to the classical approach, the additional margin variable ξ appears to be particularly natural

in our stochastic linear program:

$$\begin{aligned} \mathbf{0}' \cdot \mathbf{y} - 1 \cdot z + M \cdot \xi &\to \min_{\mathbf{y},z,\xi} \\ s.t. \begin{cases} \tilde{\mathbf{a}}_1' \cdot \mathbf{y} - 1 \cdot z + \xi \geq & -1, \\ (-\tilde{\mathbf{a}}_2)' \cdot \mathbf{y} - 1 \cdot z + \xi \geq & 1\,, \end{cases} \\ \text{where} \quad \tilde{\mathbf{a}}_1 \sim Q_1,\ \tilde{\mathbf{a}}_2 \sim Q_2. \end{aligned} \tag{SVM-soft}$$

Our soft-margin model has the advantage that, if we are unsure about the proper class labels of the training points, we can introduce a random coefficient for ξ that describes the level of certainty in labeling. It is also clear that we can use the kernel trick here, because the inner product and the induced norm are sufficient for all calculations in the algorithm.

Further, our robust optimization model and the new algorithm can be applied in many other fields of operations research. In particular, it is well suited to formalize problems in supply chain management, like the management of an inventory. This will be the topic of future work.

Bibliography

Acerbi, C. (2002). Spectral measures of risk: A coherent representation of subjective risk aversion. *Journal of Banking and Finance*, 26:1505–1581.

Adam, A., Houkari, M., and Laurent, J.-P. (2008). Spectral risk measures and portfolio selection. *Journal of Banking and Finance*, 32:1870–1882.

Adler, D. and Murdoch, D. (2011). *rgl: 3D visualization device system (OpenGL). R* package version 0.92.798, URL `http://CRAN.R-project.org/package=rgl`.

Aloupis, G. (2006). Geometric measures of data depth. In *Data depth: robust multivariate analysis, computational geometry, and applications, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 72, pages 147–158. American Mathematical Society.

Artzner, P., Delbaen, F., Eber, J.-M., and Heath, D. (1999). Coherent measures of risk. *Mathematical Finance*, 9(3):203–228.

Barber, C. B., Dobkin, D. P., and Huhdanpaa, H. (1996). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483.

Bazovkin, P. (2013). *PortfolioTR: Portfolio Selection with Multivariate Distortion Risk Measures. R* package version 2.0, URL `http://CRAN.R-project.org/package=PortfolioTR`.

Bazovkin, P. (2014). Geometrical framework for robust portfolio optimization. *Discussion Papers in Econometrics and Statistics*, (01/14). Institute of Econometrics and Statistics, University of Cologne.

Bazovkin, P. and Mosler, K. (2011). *WMTregions: Exact calculation of weighted-mean trimmed regions. R* package version 3.2, URL `http://CRAN.R-project.org/package=WMTregions`.

Bazovkin, P. and Mosler, K. (2012a). An exact algorithm for weighted-mean trimmed regions in any dimension. *Journal of Statistical Software*, 47(13):1–29.

Bazovkin, P. and Mosler, K. (2012b). *StochaTR: Solving stochastic linear programs with a single risk constraint*. *R* package version 1.0.4, URL `http://CRAN.R-project.org/package=StochaTR`.

Bazovkin, P. and Mosler, K. (2015). A general solution for robust linear programs with distortion risk constraints. *Annals of Operations Research*, 229(1):103–120.

Ben-Tal, A., Bertsimas, D., and Brown, D. B. (2010). A soft robust model for optimization under ambiguity. *Operations Research*, 58(4):1220–1234.

Ben-Tal, A., den Hertog, D., De Waegenaere, A., Melenberg, B., and Rennen, G. (2013). Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357.

Ben-Tal, A., El Ghaoui, L., and Nemirovski, A. (2009). *Robust Optimization*. Princeton University Press, Princeton, New Jersey.

Ben-Tal, A. and Nemirovski, A. (2000). Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical programming*, 88(3):411–424.

Bertsimas, D. and Brown, D. B. (2009). Constructing uncertainty sets for robust linear optimization. *Operations Research*, 57(6):1483–1495.

Bertsimas, D., Brown, D. B., and Caramanis, C. (2011). Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501.

Bertsimas, D. and Goyal, V. (2013). On the approximability of adjustable robust convex optimization under uncertainty. *Mathematical Methods of Operations Research*, 77(3):323–343.

Bion-Nadal, J. and Kervarec, M. (2012). Risk measuring under model uncertainty. *The Annals of Applied Probability*, 22(1):213–238.

Borgwardt, K. (2001). *Optimierung, Operations Research, Spieltheorie: Mathematische Grundlagen*. Birkhäuser Verlag, Basel.

Buckley, C. E. (1988). A divide-and-conquer algorithm for computing 4-dimensional convex hulls. In *Proceedings on International Workshop on Computational Geometry and its Applications*, pages 113–135, New York. Springer-Verlag.

Calafiore, G. C. (2007). Ambiguous risk measures and optimal robust portfolios. *Siam Journal on Optimization*, 18(3):853–877.

Cascos, I. (2007). The expected convex hull trimmed regions of a sample. *Computational Statistics*, 22:557–569.

Cascos, I. (2009). Data depth: Multivariate statistics and geometry. In Kendall, W. and Molchanov, I., editors, *New Perspectives in Stochastic Geometry*. Clarendon Press, Oxford University Press, Oxford.

Cascos, I. and Molchanov, I. (2007). Multivariate risks and depth-trimmed regions. *Finance and Stochastics*, 11:373–397.

Chazelle, B. (1993). An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10(1):377–409.

Chen, W. and Sim, M. (2009). Goal-driven optimization. *Operations Research*, 57(2):342–357.

Chen, W., Sim, M., Sun, J., and Teo, C.-P. (2010). From cvar to uncertainty set: Implications in joint chance-constrained optimization. *Operations Research*, 58:470–485.

Costa, O. L. V. and Paiva, A. C. (2002). Robust portfolio selection using linear-matrix inequalities. *Journal of Economic Dynamics and Control*, 26(6):889–909.

Cousin, A. and Di Bernardino, E. (2013). On multivariate extensions of value-at-risk. *Journal of Multivariate Analysis*, 119:32–46.

Delage, E. and Ye, Y. (2010). Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research*, 58(3):595–612.

Delbaen, F. (2002). Coherent risk measures on general probability spaces. In *Advances in Finance and Stochastics*, pages 1–37. Springer-Verlag, Berlin.

Drapeau, S. and Kupper, M. (2013). Risk preferences and their robust representation. *Mathematics of Operations Research*, 38(1):28–62.

Dyckerhoff, R. (2000). Computing zonoid trimmed regions of bivariate data sets. In Bethlehem, J. and van der Heijden, P., editors, *COMPSTAT 2000. Proceedings in Computational Statistics*, pages 295–300. Physica-Verlag, Heidelberg.

Dyckerhoff, R. (2002). Inference based on data depth. Chapter 5, in K Mosler, *Multivariate Dispersion, Central Regions and Depth: The Lift Zonoid Approach*, Springer-Verlag, New York.

Dyckerhoff, R. (2004). Data depths satisfying the projection property. *Allgemeines Statistisches Archiv*, 88:163–190.

Dyckerhoff, R., Koshevoy, G., and Mosler, K. (1996). Zonoid data depth: Theory and computation. In Pratt, A., editor, *COMPSTAT 1996. Proceedings in Computational Statistics*, pages 235–240, Heidelberg. Physica-Verlag.

Dyckerhoff, R. and Mosler, K. (2011). Weighted-mean trimming of multivariate data. *Journal of Multivariate Analysis*, 102:405–421.

Dyckerhoff, R. and Mosler, K. (2012). Weighted-mean trimming of a probability distribution. *Statistics and Probability Letters*, 82:318–325.

Edelsbrunner, H. (1987). *Algorithms in Combinatorial Geometry*. Springer-Verlag, Heidelberg.

Ekeland, I., Galichon, A., and Henry, M. (2012). Comonotonic measures of multivariate risks. *Mathematical Finance*, 22(1):109–132.

El Ghaoui, L., Oks, M., and Oustry, F. (2003). Worst-case value-at-risk and robust portfolio optimization: A conic programming approach. *Operations Research*, 51(4):543–556.

Fabozzi, F. J., Huang, D., and Zhou, G. (2010). Robust portfolios: contributions from operations research and finance. *Annals of Operations Research*, 176:191–220.

Föllmer, H. and Schied, A. (2004). *Stochastic Finance: An Introduction in Discrete Time*. Walter de Gruyter, Berlin.

Frittelli, M. and Gianin, E. R. (2002). Putting order in risk measures. *Journal of Banking & Finance*, 26(7):1473–1486.

Fukuda, K. and Rosta, V. (2004). Exact parallel algorithms for the location depth and the maximum feasible subsystem problems. In *Frontiers in global optimization*, pages 123–133. Springer US.

Gabrel, V., Murat, C., and Thiele, A. (2014). Recent advances in robust optimization: An overview. *European Journal of Operational Research*, 235(3):471–483.

Garlappi, L., Uppal, R., and Wang, T. (2007). Portfolio selection with parameter and model uncertainty: A multi-prior approach. *Review of Financial Studies*, 20(1):41–81.

Gigliarano, C. and Mosler, K. (2009). Constructing indices of multivariate polarization. *The Journal of Economic Inequality*, 7(4):435–460.

Gilbert, E. G., Johnson, D. W., and Keerthi, S. S. (1988). A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE J. Robotics Automation*, RA-4:193–203.

Goldfarb, D. and Iyengar, G. (2003). Robust portfolio selection problems. *Mathematics of Operations Research*, 28(1):1–38.

Grünbaum, B. (2003). *Convex Polytopes*. Springer-Verlag, New York, 2nd edition.

Hallin, M., Paindaveine, D., and Šiman, M. (2010). Multivariate quantiles and multiple-output regression quantiles: From l_1 optimization to halfspace depth. *Annals of Statistics*, 2:635–669. With discussion.

Hamel, A. and Heyde, F. (2010). Duality for set-valued measures of risk. *SIAM Journal on Financial Mathematics*, 1(1):66–95.

Hamel, A. H., Heyde, F., and Rudloff, B. (2011). Set-valued risk measures for conical market models. *Mathematics and Financial Economics*, 5(1):1–28.

Hamel, A. H., Löhne, A., and Rudloff, B. (2014). Benson type algorithms for linear vector optimization and applications. *Journal of Global Optimization*, 59(4):811–836.

Hamel, A. H., Rudloff, B., and Yankova, M. (2013). Set-valued average value at risk and its computation. *Mathematics and Financial Economics*, 7(2):229–246.

Holz, H. and Mosler, K. (1994). An interactive decision procedure with multiple attributes under risk. *Annals of Operations Research*, 52:151–170.

Huang, D., Zhu, S. S., Fabozzi, F. J., and Fukushima, M. (2008). Portfolio selection with uncertain exit time: A robust cvar approach. *Journal of Economic Dynamics & Control*, 32(2):594–623.

Huber, P. J. (1981). *Robust statistics*. Wiley, New York.

Jorion, P. (2006). *Value at Risk: The New Benchmark for Managing Financial Risk*. McGraw-Hill, New York, 3rd edition.

Jouini, E., Meddeb, M., and Touzi, N. (2004). Vector-valued coherent risk measures. *Finance and Stochastics*, 8(4):531–522.

Kall, P. and Mayer, J. (2010). *Stochastic Linear Programming. Models, Theory, and Computation*. Springer-Verlag, New York, 2nd edition.

Kan, R. and Zhou, G. F. (2007). Optimal portfolio choice with parameter uncertainty. *Journal of Financial and Quantitative Analysis*, 42(3):621–656.

Kawas, B. and Thiele, A. (2011). A log-robust optimization approach to portfolio management. *OR Spectrum*, 33(1):207–233.

Kay, T. L. and Kajiya, J. T. (1986). Ray tracing complex scenes. *SIGGRAPH Comput. Graph.*, 20:269–278.

Kirilyuk, V. S. (2008). Polyhedral coherent risk measures and investment portfolio optimization. *Cybernetics and Systems Analysis*, 44(2):250–260.

Knuth, D. (1997). *The Art Of Computer Programming*, volume 1. Addison-Wesley, Boston, 3rd edition.

Konno, H. and Koshizuka, T. (2005). Mean-absolute deviation model. *Iie Transactions*, 37(10):893–900.

Koshevoy, G. and Mosler, K. (1997). Zonoid trimming for multivariate distributions. *Annals of Statistics*, 25(5):1998–2017.

Koshevoy, G. and Mosler, K. (2007). Multivariate lorenz dominance based on zonoids. *AStA - Advances in Statistical Analysis*, 91:57–76.

Kusuoka, S. (2001). On law invariant coherent risk measures. *Advances in Mathematical Economics*, 3:83–95.

Lang, D. T., Swayne, D., Wickham, H., and Lawrence, M. (2010). *rggobi: Interface between R and GGobi*. *R* package version 2.1.16, URL `http://CRAN.R-project.org/package=rggobi`.

Lawrence, M. and Lang, D. T. (2010). *RGtk2: R bindings for Gtk 2.8.0 and above*. *R* package version 2.12.18, URL `http://CRAN.R-project.org/package=RGtk2`.

Liu, R. Y., Parelius, J. M., and Singh, K. (1999). Multivariate analysis by data depth: Descriptive statistics, graphics and inference. *Annals of Statistics*, 27(3):783–858. With discussion.

Liu, X., Mosler, K., and Mozharovskyi, P. (2014). Fast computation of Tukey trimmed regions in dimension $p > 2$. *ArXiv e-prints*, ArXiv:1412.5122.

López-Pintado, S. and Romo, J. (2007). Depth-based inference for functional data. *Computational Statistics & Data Analysis*, 51(10):4957–4968.

Maccheroni, F., Marinacci, M., and Rustichini, A. (2006). Ambiguity aversion, robustness, and the variational representation of preferences. *Econometrica*, 74(6):1447–1498.

Markowitz, H. (1952). Portfolio selection. *Journal of Finance*, 7(1):77–91.

Meucci, A. (2009). *Risk and asset allocation*. Springer-Verlag, Berlin Heidelbergs.

Miller, K., Ramaswami, S., Rousseeuw, P., Sellarès, J. A., Souvaine, D., Streinu, I., and Struyf, A. (2003). Efficient computation of location depth contours by methods of computational geometry. *Statistics and Computing*, 13(2):153–162.

Mosler, K. (2002). *Multivariate Dispersion, Central Regions and Depth: The Lift Zonoid Approach*. Springer-Verlag, New York.

Mosler, K. (2004). Introduction: The geometry of data. *Allgemeines Statistisches Archiv*, 88(2):133–135.

Mosler, K. (2013). Depth statistics. In Becker, C., Fried, R., and Kuhnt, S., editors, *Robustness and Complex Data Structures*, pages 17–34. Springer-Verlag, Berlin Heidelberg.

Mosler, K. and Bazovkin, P. (2014). Stochastic linear programming with a distortion risk constraint. *OR Spectrum*, 36(4):949–969.

Mosler, K. and Hoberg, R. (2006). Data analysis and classification with the zonoid depth. In Liu, R. Y., Serfling, R., and Souvaine, D., editors, *Data Depth: Robust Multivariate Analysis, Computational Geometry and Applications*, pages 49–59. American Mathematical Society.

Mosler, K., Lange, T., and Bazovkin, P. (2009). Computing zonoid trimmed regions in dimension $d > 2$. *Computational Statistics & Data Analysis*, 53:2500–2510.

Natarajan, K., Pachamanova, D., and Sim, M. (2008). Incorporating asymmetric distributional information in robust value-at-risk optimization. *Management Science*, 54(3):573–585.

Natarajan, K., Pachamanova, D., and Sim, M. (2009). Constructing risk measures from uncertainty sets. *Operations Research*, 57(5):1129–1141.

Nemirovski, A. and Shapiro, A. (2006). Convex approximations of chance contrained programs. *SIAM Journal on Optimization*, 17:969–996.

Pflug, G. C. (2006). On distortion functionals. *Statistics & Decisions*, 24(1):45–60.

Pinar, M. C. (2007). Robust scenario optimization based on downside-risk measure for multi-period portfolio selection. *OR Spectrum*, 29(2):295–309.

Prékopa, A. (1995). *Stochastic Programming*. Kluwer Academic Publishers, Dordrecht, Boston.

Rachev, S., Ortobelli Lozza, S., Stoyanov, S., and Fabozzi, F. J. (2008). Desirable properties of an ideal risk measure in portfolio theory. *International Journal of Theoretical and Applied Finance*, 11(1):19–54.

Rockafellar, R. T. (1997). *Convex analysis*. Princeton University Press, Princeton, New Jersey.

Rockafellar, R. T. and Uryasev, S. (2000). Optimization of conditional value-at-risk. *Journal of Risk*, 2:21–41.

Rockafellar, R. T., Uryasev, S., and Zabarankin, M. (2006). Optimality conditions in portfolio analysis with general deviation measures. *Mathematical Programming*, 108(2-3):515–540.

Rousseeuw, P. J. and Ruts, I. (1996). Algorithm as 307. bivariate location depth. *Applied Statistics*, 45:516–526.

Rousseeuw, P. J. and Struyf, A. (1998). Computing location depth and regression depth in higher dimensions. *Statistics and Computing*, 8:193–203.

Rüschendorf, L. (2006). Law invariant convex risk measures for portfolio vectors. *Statistics and Decisions*, 24(1):97–108.

Rüschendorf, L. (2010). Risk measures for portfolio vectors and allocation of risks. In Bol, G., Rachev, S., and Würth, R., editors, *Risk Assessment. Decisions in Banking and Insurance*, pages 133–164. Physica-Verlag, Heidelberg.

Rüschendorf, L. (2013). *Mathematical Risk Analysis*. Springer-Verlag, Berlin Heidelberg.

Ruts, I. and Rousseeuw, P. J. (1996). Computing depth contours of bivariate point clouds. *Computational Statistics and Data Analysis*, 23:153–168.

Schrage, C. (2015). Scalar representation and conjugation of set-valued functions. *Optimization*, 64(2):197–223.

Schrage, C. and Löhne, A. (2013). An algorithm to solve polyhedral convex set optimization problems. *Optimization*, 62(1):131–141.

Serfling, R. (2006). Depth functions in nonparametric multivariate inference. In Liu, R., Serfling, R., and Souvaine, D., editors, *Data Depth: Robust Multivariate Analysis, Computational Geometry and Applications*, pages 1–16. American Mathematical Society.

Sharpe, W. F. (1966). Mutual fund performance. *Journal of Business*, 39:119–138.

Shen, R. J. and Zhang, S. Z. (2008). Robust portfolio selection based on a multi-stage scenario tree. *European Journal of Operational Research*, 191(3):864–887.

Swart, G. (1985). Finding the convex hull facet by facet. *Journal of Algorithms*, 6(1):17–48.

Tukey, J. W. (1975). Mathematics and picturing data. In James, R., editor, *Proceedings of the 1974 International Congress of Mathematicians, Vancouver*, volume 2, pages 523–531.

Tütüncü, R. H. and Koenig, M. (2004). Robust asset allocation. *Annals of Operations Research*, 132(1-4):157–187.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley, New York.

Wang, S. S., Young, V. R., and Panjer, H. H. (1997). Axiomatic characterization of insurance prices. *Insurance: Mathematics and Economics*, 21(2):173–183.

Winder, R. (1966). Partitions of n-space by hyperplanes. *SIAM Journal on Applied Mathematics*, 14(4):811–818.

Zhu, X., Ding, H., and Xiong, Y. (2001). Pseudo minimum translation distance between convex polyhedra (i). *Science in China*, 44(2):217–224.

Zuo, Y. and Serfling, R. (2000). Structural properties and convergence results for contours of sample statistical depth functions. *Annals of Statistics*, 28:483–499.

List of Figures

Symbols

$\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ ...	real vectors	All chapters
$(x_1, \ldots, x_d)'$	components of a real vector $\mathbf{x} \in \mathbb{R}^d$	All chapters
$\mathbf{0}$	vector of zeros	All chapters
$\mathbf{1}$	vector of ones	All chapters
$\mathbf{A}, \mathbf{C}$ ...	real matrices	All chapters
$C(\mathbf{v})$	direction cone for a vertex $\mathbf{v}$	Chapter 2
D_α	trimmed region of depth α	All chapters
ZD_α	zonoid region of depth α	Chapters 2 and 3
$D_{\mathbf{w}_\alpha}$	WM region with a weight vector $\mathbf{w}_\alpha$	Chapters 4, 5, 6
$\tilde{\mathbf{a}}$, $\tilde{\mathbf{b}}$ ...	random vectors	Chapters 5 and 6
$\tilde{\mathbf{A}}, \tilde{\mathbf{B}}$ ...	random matrices	Chapters 5 and 6
Δ^d	unit simplex in $\mathbb{R}^d$	Chapters 4 and 5
$\boldsymbol{\omega}$	portfolio vector	Chapter 4
μ^m	set-valued risk measure in $\mathbb{R}^m$	Chapters 4, 5, 6
ρ^d	vector- or scalar-valued risk measure in $\mathbb{R}^d$	Chapters 4, 5, 6
$\mathcal{F}$	admissable set	Chapters 4 and 6
$\mathcal{X}$	feasible set	Chapters 5 and 6
$\mathcal{U}$	uncertainty set	Chapters 5 and 6
$\mathcal{G}$	convolution set	Chapter 6

Curriculum Vitae

Personal details

Name	Pavel Bazovkin (Pavlo Bazovkin)
Date of birth	19th of June, 1986
Birthplace	Simferopol, Russia
E-mail	Lift.zonoid@yandex.com

Education and Research

Before 2002	**Klovsky Lyceum, Kiev, Ukraine** Graduation with Gold medal
2002 – 2006	**National Technical University of Ukraine "Igor Sikorsky Kiev Polytechnic Institute", Ukraine** Diploma with Distinction Degree: Bachelor of Computer Science
2006 – 2008	**National Technical University of Ukraine "Igor Sikorsky Kiev Polytechnic Institute", Ukraine** Diploma with Distinction Degree: Master of Computer Science
2008 – 2014	**University of Cologne, Germany** Grantee of the German Research Foundation (DFG); Research assistant at the Institute of Statistics and Econometrics

Research interests	Computational and applied statistics, non-parametric data analysis and machine learning, operations research, stochastic and robust optimization, mathematical finance and risk management
Languages	English (fluent), German (fluent), French (good command), Russian (native speaker), Ukrainian (good command)