UNIVERSITEIT GENT

FACULTY OF BIOSCIENCE ENGINEERING

Department of Mathematical Modelling, Statistics and Bioinformatics

Mathematical optimization methods for the analysis of compositional data: subset selection, unmixing and prediction

ir. Jan Verwaeren

Thesis submitted in fulfillment of the requirements for the degree of
Doctor (Ph.D.) of Applied Biological Sciences

Academic year 2013-2014

Supervisor:	Prof. dr. Bernard De Baets Department of Mathematical Modelling, Statistics and Bioinformatics Ghent University, Belgium
Examination committee:	Prof. dr. ir. Frank Devlieghere (Chairman) Dr. Krysztof Dembczyński Prof. dr. Patrick De Causmaecker Prof. dr. Hans De Meyer Prof. dr. ir. Veerle Fievez Prof. dr. Willem Waegeman Prof. dr. ir. Olivier Thas
Dean:	Prof. dr. ir. Guido Van Huylenbroeck
Rector:	Prof. dr. Anne De Paepe

ir. Jan Verwaeren

Mathematical optimization methods for the analysis of compositional data: subset selection, unmixing and prediction

Thesis submitted in fulfillment of the requirements for the degree of

Doctor (Ph.D) of Applied Biological Sciences

Academic year 2013-2014

Dutch translation of the title:
Wiskundige optimalisatiemethoden voor de analyse van compositionele data: subset-selectie, ontmenging en voorspelling

Please refer to this work as follows:
Jan Verwaeren (**2014**). *Mathematical optimization methods for the analysis of compositional data: subset selection, unmixing and prediction*, PhD Thesis, Department of Mathematical Modelling, Statistics and Bioinformatics, Ghent University, Ghent, Belgium.

ISBN 978-90-5989-711-3

Acknowledgements - Dankwoord

Ik denk dat een doctoraatsthesis in de eerste plaats het resultaat is van de inspanningen van een heel team, veeleer dan van een enkel individu. Het voordeel van de auteur te zijn, is dat je veruit het grootste deel van de lofbetuigingen in ontvangst mag nemen. Daarom wil ik hier toch de belangrijkste mensen in 'mijn team' bedanken.

In de eerste plaats wil ik mijn ouders, Theo en Wivina, en mijn vriendin, Ilse, bedanken. Mijn ouders hebben me vanaf de eerste dag die ik me kan herinneren tot vandaag steeds gesteund en hebben me de mogelijkheid gegeven om verder te studeren. Ook tijdens mijn doctoraatsonderzoek kon ik steeds op hen rekenen. Mijn lieve partner Ilse heeft me de voorbije zeven jaar ongelooflijk hard gesteund. Voor mij is zij dan ook een van de belangrijkste 'co-auteurs' van dit werk. Tenslotte wil ik hen allemaal, en ook mijn schoonouders Paul en Rita en mijn zus Leen bedanken om mijn eerste dertig levensjaren te vullen met aangename momenten, een onmisbare randvoorwaarde voor dit werk.

Mijn promotor prof. Bernard De Baets wil ik bedanken voor het vertrouwen dat hij in mij gesteld heeft de voorbije zeven jaar. Door zijn positieve en enthousiaste houding tegenover mijn onderzoek heb ik steeds met plezier aan mijn doctoraat gewerkt. Ik heb van Bernard heel veel geleerd, temeer omdat zijn deur steeds voor mij open stond.

In de zeven jaar waarin ik aan mijn doctoraat gewerkt heb, heb ik een heleboel fijne collega's ontmoet (als ik even nadenk zijn er dat aan de vakgroep alleen al zo'n 40). Velen onder hen hebben rechtstreeks of onrechtstreeks iets bijgedragen tot mijn doctoraatsthesis. Een aantal onder hen wil ik graag expliciet bedanken. In de eerste plaats Willem, van wie ik ongelooflijk veel heb bijgeleerd en die een drijvende kracht was achter het vierde deel van mijn doctoraatsthesis. Daarnaast wil ik ook Michael bedanken. De ideeën die schuilgaan achter het derde deel van mijn doctoraatsthesis zou ik nooit gehad hebben zonder de samenwerkingen die hij heeft uitgebouwd. Tenslotte wil ik ook nog al mijn collega's van de onderzoekseenheid Kermit danken en in het bijzonder mijn bureaugenoten Arne, Pieter, Steffie en Jan. Dankzij hen ben ik de voorbije zeven jaar steeds met plezier komen werken.

Jan Verwaeren
Gent 10/06/2014

Table of contents

Abbreviations

ACO Ant Colony Optimization

AS Ant System

B & B branch-and-bound

BSR bit string representation

ilr isometric log-ratio

KKT Karush-Kuhn-Tucker

LICQ linear independence constraint qualification

LMM linear mixture model

MKLR multi-class kernel logistic regression

MMAS Max-Min Ant System

NIR near infra red

POC proportional odds model with ordered components

RBF radial basis function

RF Random Forests

RKHS reproducing kernel Hilbert space

SOCP second order cone program

1 Introduction

1.1. A general overview

In most areas of applied research and in industry, scientists and engineers are —from time to time— confronted with mixtures. As a result, mixtures appear in numerous (research) questions or engineering problems. As a starting point, we give a typical example of such a problem. This example will reappear several times throughout this introductory chapter.

Consider the characterization of a soil sample by means of the fractions of each soil separate (sand, silt and clay) present in that sample. Using this simple characterization, a soil sample can be represented as a mixture of sand, silt and clay. A soil scientist might be interested in the following question: What is the relationship between the depth at which a sample is taken and the fractions of sand, silt and clay in the sample?

To answer the question that is raised in the example given above, a traditional scientific methodology requires that: firstly, the loosely formulated objective is translated into a formal objective (in the example this could for instance be a formal hypothesis); secondly, data is collected that can be used to assist in reaching the objective; thirdly, some conclusions are drawn based on a (statistical) analysis of the data; and fourthly, based on the conclusions, some actions are taken.

In this dissertation, we will mainly be focusing on the third step of this process. To perform an analysis of the data, a data-analyst typically has a variety of traditional statistical tools at his disposal. Interestingly, there are several reasons (we elaborate on this later) why most common statistical procedures cannot be used to analyze data that is obtained from the analysis of mixtures. Instead, a data-analyst might resort to the field of *compositional data analysis*, which is a subfield of statistics that is devoted to the analysis of compositional data[1]. However, even though the inadequacy of traditional statistics for the analysis of compositional data has been known for more than a century, most work within the field of compositional data analysis has mainly focused on the generalization of traditional statistical techniques to compositional data. On the other hand, several problems that naturally arise when analyzing data of mixtures, such as data selection, uncertainty propagation and predictive modeling, have not been dealt with extensively. In this dissertation,

[1] When we refer to an object as being a mixture, we typically express our interest in its composition. More precisely, the composition of such an object is generally described by an (exhaustive) enumeration of its components and a vector of values that represent the relative contribution of each of these components to the mixture. Such a vector is called a compositional vector or shortly, a *composition*. Compositional data is data that consists of compositional vectors.

we will develop data-analysis techniques that can be used to provide an answer to these problems.

The success of numerous modern data-analysis techniques relies upon the fruitful combination of numerical optimization, problem knowledge and some basic insights in data analysis. As an overall approach in this dissertation, we will translate several questions that can arise when data of mixtures are analyzed into mathematical optimization problems. The use of domain knowledge (including the fact that we are working with mixtures) and several well-known insights in data analysis, mainly from the field of machine learning, will allow us to define optimization problems that effectively capture the essence of the problems that we wish to solve. Moreover, it will turn out that most of the optimization problems that are obtained are special cases of more general, well-known optimization problems. The advantage of this is twofold: firstly, it allows us to use the extensive literature on numerical optimization as well as some powerful numerical solvers to solve the resulting optimization problems; and, secondly, the modifications of existing numerical recipes that we propose can be useful in the more general settings as well.

1.2. A road-map to this dissertation

This dissertation consists of one introductory part (Part I), three main parts (Parts II–IV) and one concluding part (Part V). This structure is visualized in Figure 1.1. This dissertation can be positioned at the crossroads of the fields of compositional data analysis and mathematical optimization. To assist the reader, Part I contains two chapters that introduce these fields, and present several definitions and well-known results. The reader is strongly encouraged to read the introductory chapter on compositional data analysis. The reader who is less familiar with mathematical optimization (for example: KKT conditions, conic programming or branch-and-bound algorithms) is encouraged to read the introductory chapter on mathematical optimization.

Parts II–IV contain the main contributions of this dissertation. Each of these parts focuses on a limited number of research objectives at the crossroads of compositional data analysis and mathematical optimization. Therefore, these parts build upon the introductory material that is presented in Chapters 2 & 3. However, Parts II–IV can be read independently. The first chapter of each of these parts is (to some extent) meant to introduce and motivate the following chapters. Even though these 'introductory' chapters contain mostly novel material, the technical details are kept to a minimum.

Part V summarizes the most important results in this dissertation, and provides suggestions for further research. It should be noted, however, that most of the

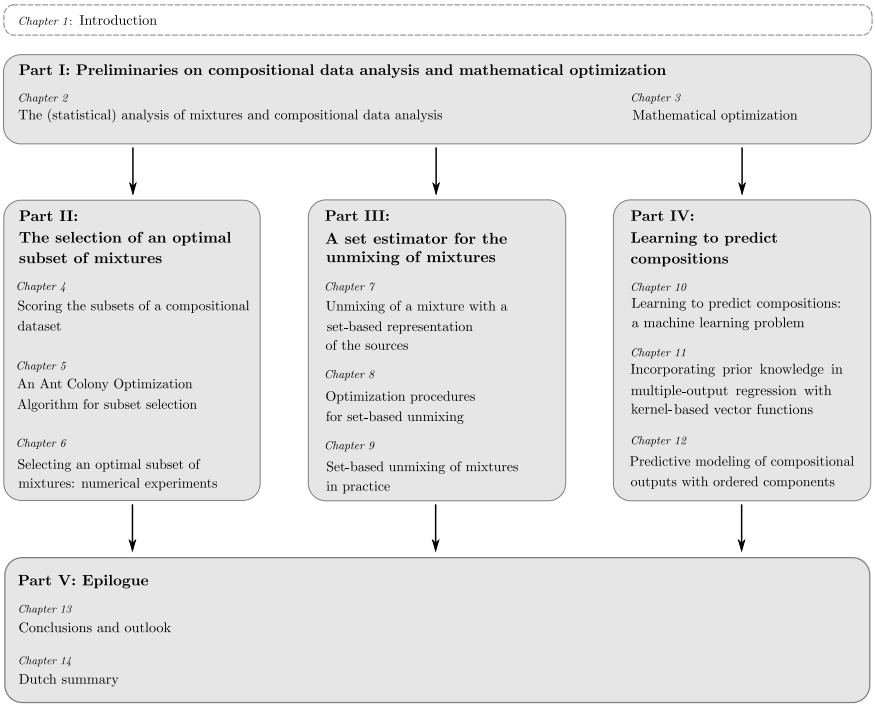


Figure 1.1: A road-map to this dissertation.

discussions in this dissertation are interwoven into the main content (mostly due to their technical nature).

Throughout this dissertation, we will adhere to a set of **notational conventions**. These conventions are summarized in Appendix 1.A at the end of this chapter. In the remainder of this chapter, the three main parts are briefly introduced. Moreover, for each of the main parts, the key research objectives are formulated.

1.3. The selection of an optimal subset of mixtures (Part II)

1.3.1. Problem setting

Modern high-throughput measuring equipment allows researchers to analyze large numbers of samples in a limited amount of time. The use of this equipment for the analysis of the (chemical) composition of mixtures often results in large databases that contain compositional data. Even though the acquisition of these databases is generally interesting, it is often only a single step of an entire research project.

In some cases, the post-processing of these samples (or the data obtained from them) is more expensive. For instance, a further analysis might call for a complex, time-consuming laboratory analysis, or the analysis of the resulting dataset needs to be performed by a numerical procedure that scales badly with the size of the dataset. Here, due to budgetary or time constraints, a researcher is sometimes forced to select a subset from the original set of mixtures. This problem setting naturally raises the question of how to select such a subset.

Part II of this dissertation focuses on the selection of an optimal subset of mixtures. The goal of the second part of this dissertation is threefold.

Objective II.1: *The translation of the mixture subset selection problem into a number of formal mathematical optimization problems.*

Objective II.2: *The development of an Ant Colony algorithm that is capable of solving these optimization problems, as well as a theoretical analysis and generalization of the proposed procedure.*

Objective II.3: *The illustration of the proposed methodology by means of a case study.*

The three objectives above constitute the main topics of Chapters 4, 5 and 6 of the second part of this dissertation. In the following section, the objectives of these chapters are described in detail.

1.3.2. A brief overview of Part II

Scoring the subsets of a compositional dataset (Chapter 4)

Inevitably, the reduction of a dataset to a subset will imply a loss of information. However, we can attempt to select a subset that minimizes the loss of information that is relevant w.r.t. a specific research goal. More precisely, a score function can be designed that scores each subset in terms of the amount of relevant information that it contains. Subsequently, the subset that maximizes this function can be identified by solving an optimization problem. Even though this approach is fairly straightforward, its successful application requires two key ingredients: (1) a good score function (which is the main topic of Chapter 4), and (2) a procedure that is able to select the subset that maximizes this score function (which is the main topic of Chapter 5).

In general, a good score function should reflect the amount of relevant information contained in a subset of a given dataset. Several classes of score functions can be constructed that only rely on the distances between the elements of the dataset. Within these classes of score functions, the selection of a good score function boils

down to the selection of a good distance measure. However, due to some of the characteristics of compositional data, most of the well-known distance measures (such as for instance Euclidean distance) are not very meaningful. Instead we resort to more specialized distance measures from the field of compositional data analysis.

In Chapter 4, the approach that is briefly described above will be used to translate the mixture subset selection problem into a formal mathematical optimization problem. As stated earlier, the selection of a score function is essential to obtain informative subsets. Several classes of score functions are introduced and studied.

An Ant Colony Optimization Algorithm for subset selection (Chapter 5)

Interestingly, the problem that is addressed in the first part of this dissertation belongs to the class of subset selection problems (that includes well-known problems such as the knapsack problem). Unfortunately, a lot of well-known subset selection problems are extremely hard to solve within a reasonable amount of time. Instead, one often needs to resort to heuristics to find a potentially suboptimal solution. In Chapter 5, an Ant Colony Optimization procedure (which is a meta-heuristic) is developed, which can be used to optimize the score functions that were proposed in Chapter 4. Even though the proposed procedure is a heuristic, several interesting (theoretical) properties regarding its functioning can be shown. Moreover, these results show that the basic philosophy of the proposed procedure can be employed to address a more general class of optimization problems.

Selecting an optimal subset of mixtures: numerical experiments (Chapter 6)

In Chapter 6, a real-life problem setting in agriculture is considered that requires the selection of a subset of a set of mixtures. The Ant Colony Optimization procedure that is developed in Chapter 5 is used to find a subset that optimizes one of the score functions that are described in Chapter 4. This real-life problem setting serves as a test-case for our Ant Colony Optimization procedure. We compare our novel procedure with several competitors.

Moreover, we study the difference between the classes of score functions that were introduced in Chapter 4 in a practical setting. As argued in Chapter 4, score functions can either focus on subsets that are representative for the data (a first class) or subsets that exert maximal variability (a second class). Consequently, it can be argued that these classes of score functions will focus on different kinds of subsets. In Chapter 6, the influence of this phenomenon on the subset that is selected is studied by means of a practical case study.

From the introductory chapter on compositional data analysis, we know that the compositional nature of the data requires specific distance measures to be used.

In this chapter, it is illustrated that ignoring the compositional nature of the data (by using for instance the Euclidean distance) has a strong influence on the characteristics of the subsets that are selected.

1.4. A set estimator for the unmixing of mixtures (Part III)

1.4.1. Problem setting

When we refer to an object as being a mixture, we typically express our interest in its composition. More precisely, the composition of such an object is generally described by an (exhaustive) enumeration of its components and a vector of values that describes the relative contribution of each of these components to the mixture. In Part III of this dissertation, we will be focusing on the problem of predicting the relative contribution of one of these components to that mixture.

As a starting point, consider the following example. "Assume you have been given a cup of water with a salinity of $y = 21$ ppt (parts per thousand). Moreover, it is told that the water in this cup is a blend of fresh water (salinity (c_1) of 2 ppt) and sea water (salinity c_2 of 40 ppt). Subsequently, you are asked to estimate the proportional amount x_1 of fresh water (ppt) in the cup." Accepting some very natural rules regarding the way mixtures are formed (we elaborate on this in Part III), it is trivial to see[2] that a relative amount of $x_1 = 50\%$ is the only answer that respects the data in this problem. The problem that is described here is sometimes referred to as the unmixing of a mixture.

In this example, the components (fresh and sea water) were described in a precise manner by means of their salinity (2 ppt and 40 ppt). Often, such a precise description is not available. For example, on earth, the salinity of fresh water ranges between 1 and 3 and the salinity of sea water ranges between 30 and 45. This means that, using only salinity, the 'source' sea water does not allow a precise representation (*i.e.* by means of a single value). Instead, the source sea water is represented by an interval on the salinity scale. We can interpret this interval in a possibilistic sense, *i.e.* "*for every scalar within this interval there exists at least one sea or ocean that has a salinity level that is equal to that scalar*". We say that the salinity can only provide an imprecise description of sea water and fresh water. This imprecision has an implication on our initial question (the relative amount of fresh water in the cup). Indeed, using only the imprecise information on the salinities, there are multiple answers that respect the problem data. For example, choosing $c_1 = 2$, $c_2 = 40$ (both values are *possible* according to the given intervals) we have that $x_1 = 0.5$. Alternatively, choosing $c_1 = 1.5$ and $c_2 = 42$,

[2] For $x_1 = 0.5$, we have that $0.5\,c_1 + (1 - 0.5)\,c_2 = y$.

we obtain $x_1 \approx 0.52$. As both results respect the problem data, we say that they are both possible. We conclude that there exists a set of values that are possible candidates for x_1. This contrasts the precise setting in which there is only one candidate.

In the toy example presented here, the computation of the set of possible candidates for x_1 is trivial. However, when the sources are represented by means of subsets of $\mathbb{R}^n$ (instead of $\mathbb{R}$ in the toy example) the computation of this set becomes more complicated. In Part III of this dissertation, we present a methodology that can be used to compute these candidate sets efficiently in a very general setting. Moreover, we will see that these so-called set estimators can be of interest to researchers in several domains within the life sciences.

Part III of this dissertation focuses on the unmixing of a mixture when its sources are described in an imprecise manner. The main objectives of the third part of this dissertation are enlisted hereafter.

Objective III.1: *The mathematical translation of the unmixing problem for imprecisely described sources to obtain a set estimator for the proportional contribution of a source to a mixture.*

Objective III.2: *The definition of a mathematical optimization problem that can be used to compute this set estimator in practice.*

Objective III.3: *The development of a procedure that can be used to solve the resulting optimization problem efficiently.*

Objective III.4: *The generalization of the set estimator to high-dimensional settings and settings in which sparsity is required.*

Objective III.5: *The illustration of the proposed methodology by means of a case study.*

Objectives III.1 and III.2 are the main topics of Chapter 7. Objectives III.2 and III.3 are the main topics of Chapter 8. In Chapter 9, the fifth objective is considered. In the following section, these chapters are described in detail.

1.4.2. A brief overview of Part III

Unmixing of a mixture with a set-based representation of the sources (Chapter 7)

The unmixing of mixtures, *i.e.* the computation of the proportional contribution of a set of sources to a mixture, is of interest to a multitude of applied research disciplines. As a result, there exists an extensive literature (mainly in applied

research areas) on the unmixing of mixtures. Traditional approaches require that the sources and the mixture are represented as points in a (potentially high-dimensional) space. Often, such a representation is either unavailable or overly simplifying. In those cases, we argue that the sources can sometimes be represented by means of subsets of the space considered. Those subsets should be interpreted in a possibilistic manner. A set represents a collection of possible representations of a source. In these circumstances, traditional unmixing procedures cannot be applied to obtain an estimate of the proportional contributions of the sources to the mixture.

In Chapter 7, we introduce a set estimator that can deal with the imprecise description of the sources. Moreover, we show that in practice, the set estimate of the proportional contribution of one of the sources to the mixture can be obtained by solving an optimization problem. Moreover, by means of a literature study, we illustrate the wide applicability of our set estimator.

Optimization procedures for set-based unmixing (Chapter 8)

To be able to apply the set estimator that is introduced in Chapter 7 in practice, we need to solve the optimization problem that was introduced in Chapter 7 in an efficient manner. Unfortunately, as this optimization problem is not convex, directly solving it can be hard. In Chapter 8, we propose an equivalent quasi-convex optimization problem that can be solved efficiently.

Often, the data (or information) that is used to describe the sources or the mixture is noisy. The noise that is present in the data influences the set estimate that is obtained. In Chapter 8, we study how noise influences the estimates that are obtained by our set estimator. Interestingly, the additional uncertainty that is introduced by the noise in the data that is used, can be incorporated into our set estimator in an intuitive manner. Moreover, these modifications still allow a set estimate to be computed efficiently.

Set-based unmixing of mixtures in practice (Chapter 9)

In Chapters 7 and 8, multiple set estimators have been proposed. In Chapter 9, the differences between these set estimators are studied by means of a series of experiments on artificially generated data. Moreover, by means of a real-life case study, it is illustrated that the estimators that were proposed can be useful in practice.

1.5. Learning to predict compositions (Part IV)

1.5.1. Problem setting

Often, mixtures can be represented in multiple manners. For example, the introductory problem hints at the characterization of a soil sample by means of (1) the fractions of the soil separates the sample contains or (2) the depth at which the sample was taken. We say that a mixture can be represented in two different representation spaces. In the fourth part of this dissertation, several methodologies are developed that can be used to learn a link between both representations. We approach this learning problem as an inductive inference process. More precisely, we develop several methodologies that use a dataset (containing observations of two characterizations of a set of mixtures) to learn a mapping from a first representation space (the input space) to a second representation space (the output space). We focus on settings where the output space is a space of compositions (*i.e.* a q-dimensional simplex). Subsequently, given the representation of a new mixture in the input space, the mapping that is learned will be used to predict the representation of that mixture in the output space. For example, given the depth at which a soil sample is taken, this mapping will be used to predict the fractions of the soil separates that this sample contains. Consequently, the problem of learning such a mapping from data is called a predictive modeling problem. Within the field of machine learning or statistics, the problem that was described above can be seen as a special case of the multivariate regression problem.

In the fourth part in this dissertation, we will focus on learning predictive models for compositional outputs. Therefore, this part fits within the field of machine learning. As argued before, the problem of learning a predictive model that can be used to predict the composition of a mixture can be seen as a regression problem. However, due to some specific characteristics of compositional data, traditional multivariate regression procedures cannot be applied directly. For example, the outputs represent proportional amounts, which means that they are non-negative and can be assumed to add up to one. Even though these properties are natural when studying compositional data, traditional multivariate regression methods do not take these properties into account.

Part IV of this dissertation focuses on the development and application of procedures that can be used to learn predictive models for compositional outputs from data. The main objectives of the fourth part of this dissertation are enlisted hereafter.

Objective IV.1: *The study of several loss functions that can be used when learning to predict compositions.*

Objective IV.2: *The incorporation of prior knowledge in multivariate regression problems.*

Objective IV.3: *The development of procedures for learning predictive models for compositional outputs with ordered components.*

Objectives IV.1–IV.3 constitute the main topics of Chapters 10–12. In the following section, the objectives of these chapters are described in detail.

1.5.2. A brief overview of Part IV

Learning to predict compositions: a machine learning problem (Chapter 10)

The development and application of predictive modeling methods is one of the main themes in the field of machine learning. In this first chapter, the problem of learning predictive models with compositional outputs is formalized as a machine learning problem. An essential step in this process is the choice of a specific loss function. In principle, a loss function is a function that is used to penalize prediction errors. For example, let $\mathbf{y}$ be the observed fractions of the soils separates in a soil sample and let $\hat{\mathbf{y}}$ be the predicted fractions for that sample. Mostly, we will have that $\hat{\mathbf{y}} \neq \mathbf{y}$. The loss function can be interpreted as a function that measures the dissimilarity between $\mathbf{y}$ and $\hat{\mathbf{y}}$. Naturally, for a good predictive model, this dissimilarity will be small. Therefore, we can state that learning a predictive model from a given dataset amounts to the selection of the function (from a set of candidate functions) that minimizes the dissimilarity between the predicted values and the observed values for that dataset[3]. It is not hard to see that the loss function that is chosen will strongly influence this selection. In this chapter, we will define and study different loss functions that can be used when learning predictive models with compositional outputs.

Finally, for the inexperienced reader, this chapter introduces several key concepts of machine learning that will be used throughout the fourth part of this dissertation.

Incorporating prior knowledge in multiple-output regression with kernel-based vector functions (Chapter 11)

The isometric log-ratio transform allows to transform the problem of learning a function with compositional outputs into an equivalent multiple-output regression problem (where the output space is a Euclidean space). This equivalence allows traditional multiple-output regression procedures to be used for learning predictive

[3] In that sense, the problem of learning a predictive model ultimately leads to an optimization problem.

models with compositional outputs. In Chapter 11, this path will be followed to solve learning problems with compositional outputs. It is well known that the incorporation of domain knowledge into a learning procedure can lead to an improved predictive performance. In this chapter several types of prior knowledge that can occur in a multiple-output regression setting are identified. Moreover, the development of a new matrix-valued kernel allows these types of prior knowledge to be included simultaneously into the learning problem. Theoretical results and empirical results illustrate that the resulting models have an improved predictive performance. Lastly, it is illustrated that the types of prior knowledge that were identified naturally occur in a variety of learning problems with compositional outputs.

Predictive modeling for compositional outputs with ordered components (Chapter 12)

As mentioned earlier, a composition is a vector that represents the relative contribution of a set of components to a mixture. In this chapter, we consider the special case where the components have a (natural) linear ordering. More precisely, this ordering can be seen as a type of domain knowledge that can be exploited when learning a predictive model. A general framework is presented that can be used to learn a predictive model with compositional outputs with ordered components. Moreover, the proposed methodology is validated by means of an extensive experimental section.

By relying on several assumptions, models that are specifically designed to predict ordered responses can result in an improved predictive performance. However, when these assumptions are not fulfilled, the predictive performance will deteriorate. In the second part of Chapter 12, a relaxation of the original framework is proposed. The relaxed framework can be seen as an intermediate form between the setting with linearly ordered components and the setting where the components are unordered. Alternatively, it can be seen as a methodology that is designed to predict compositions with ordered components, but makes less assumptions. Interestingly, this relaxed form gives rise to a conic optimization problem that can be solved efficiently with (specialized) existing numerical solvers.

1.A. Notational conventions

The use of mathematics to formalize and solve problems is a recurrent theme throughout this dissertation. Therefore, we will be needing a rather extensive mathematical notation. Even though several notational aspects are application specific, we will respect several conventions that are common in modern literature.

- **Sets**

  Sets are denoted as capitalized characters. For example, X will be used to denote a generic set. However, mostly, we will be using notations like $X \subset \mathbb{R}^n$ to denote that X is a proper subset of an n-dimensional real vector space. Sets can be indexed. For example, $X_1 \subseteq \mathbb{R}^n$ and $X_2 \subseteq \mathbb{R}^n$ denote two subsets of $\mathbb{R}^n$.

- **Vectors**

  Throughout this dissertation, we will often be using a vector notation. Vectors are denoted as boldface characters and are assumed to be column vectors. For example, $\mathbf{x} \in \mathbb{R}^n$ represents an $n \times 1$ column vector. Occasionally, we will write that $\mathbf{x}$ is an n-vector. The transpose of the vector $\mathbf{x}$ is denoted $\mathbf{x}^\top$ and the ith element is denoted x_i. This means we can write $\mathbf{x} = (x_1, \ldots, x_n)^\top$. Moreover, $\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^\top \mathbf{x}}$ is the L_2-norm of $\mathbf{x}$, and $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$ is the L_1-norm of $\mathbf{x}$.

  Vectors can be indexed. For example, $\mathbf{x}_1 \in \mathbb{R}^n$ and $\mathbf{x}_2 \in \mathbb{R}^n$ indicates that both $\mathbf{x}_1$ and $\mathbf{x}_2$ are $n \times 1$ vectors. Moreover, the jth element of a vector $\mathbf{x}_i \in \mathbb{R}^n$ is denoted $x_{i,j}$. On some occasions, we will use $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle = \mathbf{x}_1^\top \mathbf{x}_2$ (often to stress the geometrical interpretation of the (inner) product).

  Throughout this dissertation, several "special" vectors are used: $\mathbf{0}_n$ denotes a $n \times 1$ vector of zeros; $\mathbf{1}_n$ denotes a $n \times 1$ vector of ones.

- **Inner products and norms**

  When $\mathbf{a}$ and $\mathbf{b}$ are elements of a vector space or a Hilbert space H, we will use $\langle \mathbf{a}, \mathbf{b} \rangle_H$ to refer to the inner product of $\mathbf{a}$ and $\mathbf{b}$. Moreover, we denote $\|\mathbf{a}\|_H = \sqrt{\langle \mathbf{a}, \mathbf{a} \rangle_H}$. In the special case that H is the p-dimensional Euclidean vector space, we write $\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^\top \mathbf{b}$.

- **Matrices**

  Matrices are denoted as boldface capitalized characters. For example, $\mathbf{A} \in \mathbb{R}^{m \times n}$ denotes an $m \times n$ matrix of real numbers. The ith row of a matrix $\mathbf{A}$ is denoted $\mathbf{A}_{i,\cdot}$. The jth column of $\mathbf{A}$ is denoted $\mathbf{A}_{\cdot,j}$. Moreover, the element in the ith row of the jth column is denoted $\mathbf{A}_{i,j}$. Matrices can be indexed. For example, $\mathbf{A}^1 \in \mathbb{R}^{m \times n}$ and $\mathbf{A}^2 \in \mathbb{R}^{m \times n}$ are two $m \times n$ matrices.

$\mathbf{A}^\top$ denotes the matrix transpose of $\mathbf{A}$. For a square $n \times n$ matrix $\mathbf{A}$, $\mathbf{A}^{-1}$ denotes the inverse of $\mathbf{A}$ (assuming that its inverse exists).

The $n \times n$ identity matrix is denoted $\mathbf{I}_n$.

- **Functions**

  A generic function f with domain X and co-domain $\mathbb{R}$ is denoted $f : X \to \mathbb{R}$. Vector functions will be denoted as boldface characters. For example $\mathbf{f} : X \to \mathbb{R}^m$ represents a generic vector of m functions. Moreover, we write $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top$.

  The partial derivative of a function $f : X \to \mathbb{R}$ with respect to the variable x_i is denoted $\frac{\partial f}{\partial x_i}$. Moreover, the gradient vector of f is denoted $\nabla_{\mathbf{x}} f$ and the Hessian matrix is denoted $\nabla_{\mathbf{xx}} f$.

- **Vector (in)equalities**

  Given two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, we write $\mathbf{a} = \mathbf{b}$ if $a_i = b_i$ for $i = 1, \ldots, n$. Moreover, we write $\mathbf{a} \leq \mathbf{b}$ if $a_i \leq b_i$ for $i = 1, \ldots, n$. Finally, if $\mathbf{a} \leq \mathbf{b}$ and there is at least one index i for which $a_i \neq b_i$, we write $\mathbf{a} < \mathbf{b}$.

- **Random vectors**

  Real-valued random variables are denoted as calligraphic characters. For example, $\mathcal{X}$ is a real-valued random variable. The probability that the random variable $\mathcal{X}$ takes a value that is smaller than or equal to $x \in \mathbb{R}$ is denoted $\Pr(\mathcal{X} \leq x)$. The probability density function of $\mathcal{X}$ is denoted $\rho_{\mathcal{X}}$.

  Random variables can be indexed. For example $\mathcal{X}_1$ and $\mathcal{X}_2$ denote two random variables. Moreover, $(\mathcal{X}_1, \ldots, \mathcal{X}_n)$ is an n-dimensional random vector. When stated explicitly in the text, we can use the short-hand notation $\mathcal{X} = (\mathcal{X}_1, \ldots, \mathcal{X}_n)^\top$. The joint probability density function of the random vector $(\mathcal{X}_1, \ldots, \mathcal{X}_n)$ is denoted $\rho_{\mathcal{X}_1, \ldots, \mathcal{X}_n}$. The conditional probability density function of $\mathcal{X}_1$ given $\mathcal{X}_2$ is written $\rho_{\mathcal{X}_1 | \mathcal{X}_2}$. The expected value of a random variable $\mathcal{X}_i$ is denoted $\mathbb{E}[\mathcal{X}_i]$. Moreover, the covariance matrix of the random vector $\mathcal{X} = (\mathcal{X}_1, \ldots, \mathcal{X}_n)^\top$ is denoted as $\mathrm{cov}(\mathcal{X})$, and the variance of $\mathcal{X}_i$ is denoted as $\mathrm{var}(\mathcal{X}_i)$.

  Superscripts can be used to denote distinct random variables (or random vectors) for example, $\mathcal{X}^1$ and $\mathcal{X}^2$ are two random vectors.

PART I

PRELIMINARIES

2 The (statistical) analysis of mixtures and compositional data analysis

2.1. Introduction

When we refer to an object as being a mixture, we typically express our interest in its chemical or physical composition. More precisely, the composition of such an object is generally described by an (exhaustive) enumeration of its components and a vector of values that describes the relative contribution of each of these components to the mixture. For example, the raw analysis of three soil samples may lead to the following representation (Table 2.1):

Table 2.1: Raw analysis results of three soil samples.

	sand (g)	silt (g)	clay (g)
Sample 1	21	68	5
Sample 2	15	50	3
Sample 3	30	50	3

Here, the quantities of sand, silt and clay in the sample are represented by their respective weights. Even though these weights represent absolute quantities, they carry mainly relative information. Indeed, if the amount of soil that was analyzed would have been doubled, these quantities would have been doubled as well. This means that, unless that we are in the unlikely situation where the total amount of soil that was analyzed is important, only the relative information is relevant. Consequently, the first soil sample can equivalently be represented by the vector $(21/94, 68/94, 5/94)$, for which it holds that $21/94 + 68/94 + 5/94 = 1$, such that these numbers can be called fractions. Moreover, we call this vector a *composition*. For the second soil sample, we have the composition $(15/68, 50/68, 3/68)$. To stress the importance of such a characterization, assume that we want to compare these soil samples. Naturally, such a comparison can only be useful when it is based on the relative quantities.

The example above illustrates that, when we study mixtures, we are likely interested in studying vectors of fractions. We could for example be interested in the (statistical) relationship between the components in mixtures, or we could be interested in the influence of some external variable on this vector of fractions.

As a sub-field of statistics, the field of compositional data analysis [1] is devoted to the study of the aforementioned compositions. In this dissertation, we will be developing several tools that can be used to analyze data obtained from mixtures.

We will often be using both the dominant philosophy, as well as several (theoretical) developments from the field of compositional data analysis. Therefore, this chapter briefly introduces the field of compositional data analysis. However, it should be noted that even though the main philosophy described in this chapter will be used frequently within this dissertation, several problems that we encounter throughout this dissertation go beyond the field of traditional compositional data analysis. In Section 2.6, we elaborate on that.

The purpose of this chapter

Compositional data and its analysis will appear frequently throughout this dissertation. As we will see in this chapter, compositional data have several uncommon properties that hamper its analysis. It is often stated that compositional data carry only relative information and that any analysis procedure should respect this relative nature. This statement has far-stretching implications. Moreover, the way this statement is interpreted (or simply ignored) has a strong impact on the way that compositional data are analyzed. Mainly for the reader who is unfamiliar with the analysis of compositional data (as this field is rather small, we have generally assumed that most of the readers are not familiar with this field), this chapter introduces and illustrates several concepts that are important when analyzing such data. Therefore, this chapter can also be seen as a motivation for several design choices that are made later in this dissertation. It should be noted, however, that within this dissertation, it is not our aim to contribute to the fundamentals of compositional data analysis.

This chapter is organized as follows:

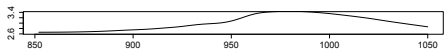
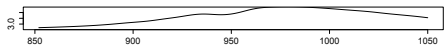
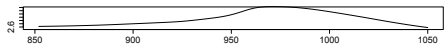
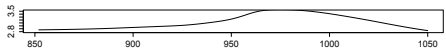
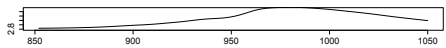
- In Sections 2.2 and 2.3, we present several examples of compositional data and introduce the most important definitions and conventions.
- In Section 2.4, we briefly review the Aitchison geometry on the simplex.
- In Section 2.5, the Dirichlet distribution is briefly described (this distribution will be used on multiple occasions).
- In Section 2.6, the importance of compositional data analysis within this dissertation is highlighted.

2.2. The essence of compositional data analysis

2.2.1. Examples of compositional data

To illustrate the broadness of the field of compositional data analysis, we start this section by giving two practical examples of settings in which compositional data analysis may be interesting. The first example, based on a publicly available

Table 2.2: Analysis results of five minced meat samples. The first column reports the sample number (conform the original dataset). The second (resp. third and fourth) column shows the relative amount (mass percentages) of water (resp. fat and protein) in the sample. The masses are expressed relative to the total amount of water, fat and protein in the sample. The fifth column shows a graphical display of the NIR-spectra of these samples (wavelength (nm) versus $\log(1/R)$). This dataset is a subset of a publicly available dataset known as the *tecator* dataset [2].

nr	water	fat	protein	NIR spectrum
1	0.607	0.226	0.167	
2	0.462	0.403	0.135	
3	0.711	0.084	0.205	
4	0.732	0.059	0.209	
5	0.587	0.257	0.156	

dataset [2], is illustrative for several applications in the applied bio-sciences and has (as far as we know) not been used in the literature on compositional data analysis. The second example is a more traditional example and is adapted from [1].

Table 2.2 reports the water, protein and fat percentages of 5 minced meat samples as well as a graphical display of their near infra-red (NIR) spectra[1]. It is clear that the objects that are studied here (the samples) fit within the description of mixtures given before. From a first inspection, it can be seen that the water, protein and fat percentages are positive and they add up to one, which reflects their proportional nature.

Compositional data analysis is not limited to the analysis of traditional mixtures. As an example, consider Table 2.3. This table reports the relative household expenditures (organized in four commodity groups) of 5 single men and 5 single women as well as the total amount spent on household expenditures. Clearly, these data are compositional. However, they can hardly be described as observations of traditional mixtures.

2.2.2. Shortcomings of traditional statistics

The main reason for the existence of the field of compositional data analysis stems from the inapplicability of traditional statistical procedures to analyze compositional data. The awareness of problems related to the analysis of data that involves

[1] This dataset is a subset of a larger (publicly available) dataset [2]. We will be using this dataset later in this dissertation

Table 2.3: Household expenditures of five single men and five single women. The first column reports the sample number (conform the original dataset). The second till the fifth column report the fraction of the total household expenditures spent on housing, food, services and other commodities. The sixth column reports the total amount that was spent (during one month, expressed in Hong Kong dollar). The last column reports the sex (M: male, F: female) of the interviewee. This table represents a subset of a dataset used in [1].

Nr	Housing	Food	Services	Other	Total (HKD)	Sex
1	0.324	0.386	0.190	0.100	1532	M
2	0.343	0.385	0.149	0.123	2448	M
3	0.238	0.390	0.174	0.199	3358	M
4	0.369	0.349	0.163	0.119	2416	M
5	0.241	0.119	0.264	0.376	6582	M
21	0.645	0.090	0.121	0.144	1271	F
22	0.648	0.261	0.070	0.021	284	F
23	0.294	0.021	0.145	0.539	3128	F
24	0.621	0.102	0.146	0.131	786	F
25	0.665	0.077	0.096	0.162	1084	F

mixtures is not new. Indeed, it is often stated that this awareness dates back to a paper by Karl Pearson (1897) [3]. Below, we illustrate several issues that arise when a traditional way of reasoning is applied to the analysis of compositional data.

Datasets such as the ones referred to above can be used to study several properties of compositions. Let us, for now, focus on the household expenditures example. We can assume that the first four columns in this dataset are i.i.d. observations of the random vector $(\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3, \mathcal{X}_4)$ that represents relative household expenditures of the population of single men. This dataset can be used to study the dependencies between the components of this random vector. A traditional statistical approach would suggest (at least as a first attempt) to use the covariance matrix as a measure of this dependency. For the dataset presented in Table 2.3 (the complete dataset was used, not merely the subset reported here), we obtain the following covariance matrix

$$\begin{pmatrix} 0.0273 & -0.0088 & -0.0141 & -0.0043 \\ -0.0088 & 0.0312 & -0.0206 & -0.0017 \\ -0.0141 & -0.0206 & 0.0309 & 0.0037 \\ -0.0043 & -0.0017 & 0.0037 & 0.0023 \end{pmatrix}$$

In this matrix, it can be seen that the majority of the sample covariances is negative. Indeed, due to the sum-to-one constraint, an increase of one component will inevitably lead to a decrease of one or more of the other components. This results in a covariance matrix that is biased. More importantly, it complicates the interpretation of such covariance structure. As the computation of covariances

as well as their interpretation is key to the vast majority of standard statistical techniques, this observation will hamper the application of these techniques.

As a second example, assume that we are interested in constructing a predictive model capable of predicting the fractions of water, protein and fat in a meat sample based on its near infra-red spectrum (this will be the topic of Part IV of this dissertation). We might be tempted to fit three separate linear regression models to the data (for instance using penalized least squares) and use these models to predict the fractions of water, protein and fat. Unfortunately, standard linear regression analysis does not guarantee that the positivity assumption of the predicted fractions will hold. Moreover, it is highly unlikely that the predicted values for the three soil separates will add up to one. As these findings are in conflict with our (natural) assumptions, we can strongly doubt whether these models are appropriate. At least, we should be cautious when drawing conclusions based on results obtained from these models.

2.2.3. Principles of compositional data analysis

According to the dominant philosophy within the field of compositional data analysis, the apparent difficulties arising when compositional data are analyzed, result from a failure to recognize the relative (or proportional) nature of these data. The main reason for this being the fact that traditional statistical procedures have been designed to operate on absolute instead of relative quantities. Because of that, most data analysis procedures process and manipulate data using the standard operations of the Euclidean vector space. Unfortunately, the relative nature of the data prevents this familiar Euclidean structure from being used. Within the field of compositional data analysis, techniques are developed that inherently use the relative nature of compositional data. More precisely, the natural characteristics of compositional data are translated in a set of principles that should be fulfilled by any procedure that is applied to compositions. As insight within these principles is essential to appreciate the philosophy of compositional data analysis (which will be used on several occasions in this dissertation), we elaborate on these principles in Section 2.4.1.

The modern manner of looking at compositional data finds its roots in the 1980s. The first landmark publication regarding this topic is probably "*The Statistical Analysis of Compositional Data*", published in the Journal of the Royal Statistical Society, by John Aitchison [4]. Moreover, the monologue "The statistical analysis of compositional data" [1] by the same author has become thé reference on the analysis of compositional data. Most of the ideas that are used nowadays are inspired on this work.

2.3. Compositional data analysis: definitions and conventions

From the examples given before, it can be deduced that a composition is a (column) vector of positive real numbers that carry only relative information regarding the contribution of a set of components (generally referred to as parts) to a mixture. Semantically, the ith element of this vector represents the proportional contribution of the ith part to the mixture. Without loss of generality, we can require that the elements of this vector add up to one. Note that, as this choice is rather arbitrary, we could as well have chosen to let the components add up to another constant. Interestingly, one of the main principles when dealing with compositional data is that the choice of this constant should not influence the results of the analysis. The following definitions closely follow the ones presented in [1].

Definition 2.1 (d-part composition)**.** *Let $d \in \mathbb{N}$, such that $d > 1$. A d-part composition is a vector $\mathbf{x} \in \mathbb{R}^d$, such that $x_i \geq 0$ for each $i = 1, \ldots, d$ and $\sum_{i=1}^{d} x_i = 1$.*

Semantically, x_i represents the proportional contribution of the ith part to the mixture.

The set of all possible d-part compositions (also called the compositional sample space) is called the d-dimensional simplex, and is formally defined hereafter.

Definition 2.2 (Simplex)**.** *The d-dimensional simplex $\mathbb{S}^d$ is the following set:*

$$\mathbb{S}^d = \left\{ \mathbf{x} \in \mathbb{R}^d \mid x_i \geq 0\,, i = 1, \ldots, d \text{ and } \sum_{i=1}^{d} x_i = 1 \right\}.$$

However, we will often limit the elements of compositional vectors to be strictly positive such that the sample space is the reduced simplex.

Definition 2.3 (Reduced simplex)**.** *The d-dimensional reduced simplex $\mathbb{S}_0^d$ is the following set:*

$$\mathbb{S}_0^d = \left\{ \mathbf{x} \in \mathbb{R}^d \mid x_i > 0\,, i = 1, \ldots, d \text{ and } \sum_{i=1}^{d} x_i = 1 \right\}.$$

Definition 2.4 (Random compositional vector)**.** *A random vector $\mathcal{X}$ whose sample space is $\mathbb{S}_0^d$ is called a d-part random compositional vector.*

Any vector $\mathbf{w} \in \mathbb{R}_+^d$ can be mapped into the reduced simplex by the following operator.

Definition 2.5 (The closure operator)**.** *The closure operator* $\mathfrak{c}$ *is the function*

$$\begin{array}{rccl} \mathfrak{c}: & \mathbb{R}_+^d & \to & \mathbb{S}^d \\ & \mathbf{w} & \mapsto & \mathbf{w}/\sum_{i=1}^d w_i\,. \end{array}$$

In the introductory example, this is exactly the operation that was applied to the data in Table 2.1.

For a given vector $\mathbf{w} \in \mathbb{R}_+^d$ let $\mathfrak{c}(\mathbf{w}) = \mathbf{x}$. Notably, for any scalar $t > 0$, we have that $\mathfrak{c}(\mathbf{w}) = \mathfrak{c}(t\,\mathbf{w}) = \mathbf{x}$. Several issues arise when traditional statistical procedures (that do not take this aspect into account) are applied to compositional data.

Lastly, we define a subcomposition as the result of applying the closure operator to a subset of the parts of a composition.

Definition 2.6 (Subcomposition)**.** *Consider a composition* $\mathbf{x} \in \mathbb{S}^d$*, and let* I *be a non-empty subset of* $\{1, \ldots, d\}$*, we call the vector* $\mathfrak{c}\left((x_i)_{i\in I}\right)$ *an* I*-subcomposition of* $\mathbf{x}$*.*

To illustrate that a subcomposition is a very natural object, consider (once more) the data in Table 2.1. For simplicity we refer to the parts sand, silt and clay respectively using the numbers 1, 2 and 3. In this example we could have decided not to measure the quantity of clay (the third column would not have been there). In this case, a soil sample is only characterized by the quantities of sand and silt. As these quantities carry only relative information, the closure operator can be used without loss of information to obtain a 2-part composition. The composition that is obtained is identical to the $\{1, 2\}$-subcomposition of the original 3-part composition.

2.4. The Aitchison geometry

2.4.1. The principles of compositional data analysis

In the literature on compositional data analysis, it is often stated that any procedure that is used to analyze compositional data should inherently respect the relative nature of the data. To formalize this (somewhat loosely formulated) requirement, three main principles have been postulated. As a general strategy, it is often recommended that procedures that are used to analyze compositional data should be validated with respect to these principles. In this section, we briefly present these three main principles of compositional data analysis. These principles were originally derived from the monograph by Aitchison [1]. However, this presentation is mainly based on a concise overview by Pawlowsky-Glahn *et al.* [5] and Aitchison [6].

Scale invariance

As a first principle, it is stated that any analysis procedure should be scale invariant. Therefore it should only use scale invariant functions. Here, a scale invariant function is a function $f : \mathbb{R}_+^d \to \mathbb{R}$ such that for all $\mathbf{x} \in \mathbb{R}_+^d$ and $\lambda > 0$ we have that $f(\mathbf{x}) = f(\lambda\,\mathbf{x})$.

This principle expresses the desired invariance of the (statistical) procedure to the measurement scale that is used. For example, if the quantities in Table 2.1 would be measured in kilograms instead of grams, the analysis should not be influenced. This property stresses the fact that compositions contain relative information.

Permutation invariance

As a second principle, it is stated that any analysis should be permutation invariant. This means that the analysis should not be affected by a permutation of the parts of the composition.

Even though this assumption may seem rather trivial at first, it is not generally satisfied. For example, to overcome the difficulty of the sum-to-one constraint (as this constraint leads to the singularity of the covariance matrix reported in Section 2.2.2) we could, once the closing operation is performed, choose to ignore the last column in our analysis. Indeed, this does not lead to any loss of information. The covariance matrix can then be computed only using the remaining columns. This would eliminate the singularity problem. However, when this procedure is repeated using a permutation of the parts, the resulting covariance matrix will be different. Any procedure that uses this covariance matrix might be affected by this permutation.

Subcompositional coherence

As a third principle, subcompositional coherence requires that erasing non-informative data should not influence the result of the analysis.

To illustrate this principle, consider the data in Tables 2.4(a) and 2.4(b). The data in both tables characterize three soil samples. In the table in panel (a) the soil samples are characterized by the proportional contributions of the three soil separates sand, silt and clay to the sample. The table in panel (b) extends this characterization by including the relative contribution of organic matter to the sample. Apart from this last column, both tables contain the same information. Stated differently, the data in the left panel is a $\{1, 2, 3\}$-subcomposition of the data in the right panel. Now, assume that a (statistical) procedure is used to characterize the dependencies between the proportions of the three soil separates (sand, silt and clay). As the organic matter content is non-informative for this characterization, the subcompositional coherence principle states that disregarding the data (panel (a) or (b)) that is used, the result should be the same.

The principle of subcompositional coherence is often stated by means of examples

Table 2.4: (a) Data from Table 2.1 after applying the closure operator (and rounding). (b) Data from Table 2.1 with an additional column representing the quantity of organic matter after applying the closure operation (and rounding).

	sand	silt	clay
S 1	0.223	0.723	0.053
S 2	0.221	0.735	0.044
S 3	0.361	0.602	0.036

(a)

	sand	silt	clay	OM
S 1	0.202	0.654	0.048	0.096
S 2	0.181	0.602	0.036	0.181
S 3	0.306	0.510	0.031	0.153

(b)

like the one given above. Even though it is often referred to as being the most important principle, we did not find a formal description of this principle. In this sense the description in [5] is worth mentioning " *Subcompositions should behave as orthogonal projections do in conventional real analysis. The size of a projected segment is less than or equal to the size of the segment itself*". This is the most formal definition of subcompositional coherence that we could find.

2.4.2. Applying the principles to traditional statistics

As a traditional statistic that is often used, consider the sample mean and the sample variance. For the relative amount of sand given in Table 2.4(a) we obtain a sample mean of 0.2683 and a variance of 0.0064. Computing these statistics using Table 2.4(b), we respectively obtain 0.2297 and 0.0045. Clearly, these values differ, which violates the subcompositional coherence principle. As a result, it is advisable not to use the sample mean and sample variance when analyzing compositional data. This conclusion should not come as a surprise, as these statistics do not take the compositional nature of the data into account.

2.4.3. Compositional data and log-ratios

To enforce the relative nature of the compositional data, it would be natural to consider ratios of the different elements of compositional vectors instead of the raw values of the elements themselves [1]. Moreover, for $\mathbf{x} \in \mathbb{S}_0^d$, we have that $x_i/x_j \in \mathbb{R}_0^+$ for $i, j = 1, \ldots, d$. Finally, as we are more familiar with working in $\mathbb{R}$, we can take logarithms of these ratios. This is the main motivation given in [1] to use log-ratios. Any $\mathbf{x} \in \mathbb{S}_0^d$ can be represented by the following matrix of log-ratios:

$$\begin{pmatrix} \ln(x_1/x_1) & \cdots & \ln(x_1/x_d) \\ \vdots & \ddots & \vdots \\ \ln(x_d/x_1) & \cdots & \ln(x_d/x_d) \end{pmatrix}$$

Using this representation, we can easily construct statistics that respect the three main principles of compositional data analysis. For example, let $\mathcal{X} = (\mathcal{X}_1, \ldots, \mathcal{X}_d)$ be a d-part compositional random vector, then we can define the following matrix:

$$\mathbf{\Gamma}_{\mathcal{X}} = \begin{pmatrix} \mathrm{var}(\ln(\mathcal{X}_1/\mathcal{X}_1)) & \cdots & \mathrm{var}(\ln(\mathcal{X}_1/\mathcal{X}_d)) \\ \vdots & \ddots & \vdots \\ \mathrm{var}(\ln(\mathcal{X}_d/\mathcal{X}_1)) & \cdots & \mathrm{var}(\ln(\mathcal{X}_d/\mathcal{X}_d)) \end{pmatrix}$$

The sum of all elements in $\mathbf{\Gamma}_{\mathcal{X}}$ can be used as a measure for the total variability of the compositional random vector. This measure (and its maximum likelihood estimator) respect the three principles. This matrix will be used in Part II of this dissertation.

Even though the matrix representation introduced above seems interesting, it contains a lot of redundancy. Indeed, this representation uses $d \times d$ real numbers to represent a $d \times 1$ vector. Moreover, these $d \times d$ real numbers cannot be seen as free variables. Due to its construction, given only a few entries of this matrix, the remaining entries trivially follow. This means that one can easily construct a matrix that cannot correspond to any compositional vector. To overcome these problems Aitchison introduced the log-ratio transform [4] and the centered log-ratio transform[1].

Definition 2.7 (Log ratio-transformation)**.** *The log-ratio transformation* $\mathfrak{l}$ *is defined by the following mapping:*

$$\begin{array}{rccl} \mathfrak{l}: & \mathbb{S}_0^d & \to & \mathbb{R}^{d-1} \\ & \mathbf{x} & \mapsto & \left(\ln\left(\frac{x_1}{x_d}\right), \ldots, \ln\left(\frac{x_{d-1}}{x_d}\right)\right) \end{array}$$

Definition 2.8 (Centered log ratio-transformation)**.** *The centered log-ratio transformation* $\mathfrak{g}$ *is defined by the following mapping:*

$$\begin{array}{rccl} \mathfrak{g}: & \mathbb{S}_0^d & \to & \mathbb{R}^{d} \\ & \mathbf{x} & \mapsto & \left(\ln\left(\frac{x_1}{g(\mathbf{x})}\right), \ldots, \ln\left(\frac{x_d}{g(\mathbf{x})}\right)\right) \end{array}$$

were $g(\mathbf{x})$ *represents the geometric mean of* $\mathbf{x}$*.*

Both transformations have advantages and disadvantages. The log-ratio transform is a bijection between $\mathbb{S}_0^d$ and $\mathbb{R}^{d-1}$. Unfortunately, this transformation is clearly not permutation invariant. The centered log-ratio transform is permutation invariant. Moreover, this transform is a bijection between $\mathbb{S}_0^d$ and $\{\mathbf{y} \in \mathbb{R}^d \mid \sum_{i=1}^d y_i = 0\}$. This means that all transformed compositions lie in a subspace of $\mathbb{R}^d$. Consequently, the sum constraint of the simplex is simply exchanged with another constraint.

Even though these transformations have several shortcomings, they have been used extensively within the field of compositional data analysis and, more importantly, form the basis for the isometric log-ratio transform (ilr-transform) [7] that is introduced hereafter.

2.4.4. A vector space for compositional data

Due to the structure of the Euclidean vector space, two vectors can be added, a vector can be multiplied with a scalar and the angle between two vectors can be computed. Unfortunately, these basic operations that data-analysts are familiar with cannot be used in the simplex. As a simple example, consider two compositions $\mathbf{x}, \mathbf{y} \in \mathbb{S}_0^d$. Using vector addition as defined on the Euclidean vector space, we have

$$\mathbf{x} + \mathbf{y} = (x_1 + y_1, \ldots, x_d + y_d)^\top .$$

Naturally, we have that $\mathbf{x} + \mathbf{y} \notin \mathbb{S}_0^d$. The same holds for scalar multiplication (we say that $\mathbb{S}_0^d$ is not closed under these operations). This means that $\mathbb{S}_0^d$ is not a (normed) vector space. Because of that, elementary operations such as computing distances cannot be performed in a mathematically sound manner. Nevertheless, the ability to compute distances (or similarities) is a necessity for most statistical procedures. In this section, we give three operations that give $\mathbb{S}_0^d$ an inner product space structure, following [8, 9].

Definition 2.9 (Perturbation)**.** *The perturbation of a composition* $\mathbf{x} \in \mathbb{S}_0^d$ *with a composition* $\mathbf{y} \in \mathbb{S}_0^d$ *is given by:*

$$\mathbf{x} \oplus \mathbf{y} = \mathfrak{c}(x_1 y_1, \ldots, x_d y_d) .$$

It is easy to see that the neutral element of $(\mathbb{S}_0^d, \oplus)$ is $\mathbf{1}_d/d$. For a given $\mathbf{x} \in \mathbb{S}_0^d$, its inverse is $\mathfrak{c}(x_1^{-1}, \ldots, x_d^{-1})$, we use $\mathbf{x}^{-1}$ to denote that inverse. Additionally, we define the operator $\ominus$ as follows: $\mathbf{x} \ominus \mathbf{y} = \mathbf{x} \oplus \mathbf{y}^{-1}$.

Definition 2.10 (Power transformation)**.** *The power transformation of a composition* $\mathbf{x} \in \mathbb{S}_0^d$ *with a scalar* $\alpha \in \mathbb{R}$ *is given by:*

$$\alpha \odot \mathbf{x} = \mathfrak{c}(x_1^\alpha, \ldots, x_d^\alpha) .$$

It can be shown [5] that $(\mathbb{S}_0^d, \oplus, \odot)$ is a vector space. However, before this space can be used in a data-analysis setting, it should be verified that the perturbation and power transformation have some intuitive properties. Even though such a motivation exists, we do not elaborate on this here. However, we mention that these operations respect the three principles defined in Section 2.4.1.

Definition 2.11 (Aitchison inner product)**.** *The inner product of* $\mathbf{x}, \mathbf{y} \in \mathbb{S}_0^d$ *is defined as:*

$$\langle \mathbf{x}, \mathbf{y} \rangle_a = \frac{1}{2d} \sum_{i=1}^{d} \sum_{j=1}^{d} \ln\left(\frac{x_i}{x_j}\right) \ln\left(\frac{y_i}{y_j}\right) .$$

The norm associated with this inner product is denoted $\|\mathbf{x}\|_a = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle_a}$. The simplex, endowed with the perturbation transformation, the power transformation and the Aitchison inner product is a normed vector space. We refer to this structure as the *Aitchison geometry.* Moreover, defining the distance function $d_a(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} \ominus \mathbf{y}\|_a$, we have that:

$$d_a(\mathbf{x}, \mathbf{y}) = \sqrt{\frac{1}{2d} \sum_{i=1}^{d} \sum_{j=1}^{d} \left(\ln\left(\frac{x_i}{x_j}\right) - \ln\left(\frac{y_i}{y_j}\right)\right)^2} .$$

From this formula it can be seen that the normed vector space is intuitively appealing (this distance function will appear several times in Part II of this dissertation). Firstly, it only uses ratios of $\mathbf{x}$ and $\mathbf{y}$, therefor the distance function uses only relative information. Moreover, the distance is defined by means of the sum of squared differences between these ratios, leading to an expression that is highly similar to the Euclidean distance.

2.4.5. A coordinate representation for compositional data

The log-ratio transformation is a bijection, as it is an isomorphism between $\mathbb{S}_0^d$ and $\mathbb{R}^{d-1}$. From a data-analysis point of view, this property could allow us to use this tranformation to: (1) express a compositional dataset as a set of points in $\mathbb{R}^{d-1}$, (2) use standard data analysis procedures, and (3) backtransform to $\mathbb{S}_0^d$. Unfortunately, this transformation is not permutation invariant. Moreover, it is not an isometry w.r.t. the Aitchinson geometry, *i.e.* $\|\mathbf{x} \ominus \mathbf{y}\|_a \neq \|\mathfrak{l}(\mathbf{x}) - \mathfrak{l}(\mathbf{y})\|_2$. Therefore, this approach is only of limited use. To overcome these difficulties, [7] defined the isometric log-ratio transformation, a transformation that is an isometry between $\mathbb{S}_0^d$ and $\mathbb{R}^{d-1}$. We present this transformation hereafter.

Let $\{\mathbf{e}_1, \ldots, \mathbf{e}_{d-1}\}$ be an orthonormal basis of $\mathbb{S}_0^d$ (w.r.t. the Aitchison geometry). A composition $\mathbf{x} \in \mathbb{S}_0^d$ can be expressed as:

$$\mathbf{x} = \bigoplus_{i=1}^{d-1} x_i^* \odot \mathbf{e}_i \,, \qquad \text{where} \qquad x_i^* = \langle \mathbf{x}, \mathbf{e}_i \rangle_a \ .$$

The vector $\mathbf{x}^* = (x_1^*, \ldots, x_{d-1}^*)^\top$ is the vector of coordinates of $\mathbf{x}$ with respect to this orthogonal basis. This transformation is generally called the isometric log-ratio

transform and was introduced in [7].

Definition 2.12 (Isometric log-ratio transformation)**.** *Given an orthonormal basis $E = \{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$ of $\mathbb{S}_0^d$, the isometric log-ratio transformation is the following mapping:*

$$\begin{array}{rccl} \mathfrak{i}_E : & \mathbb{S}_0^d & \to & \mathbb{R}^{d-1} \\ & \mathbf{x} & \mapsto & \left(\langle \mathbf{x}, \mathbf{e}_1 \rangle_a, \ldots, \langle \mathbf{x}, \mathbf{e}_{d-1} \rangle_a\right)^\top \end{array}$$

It can be shown that $\mathfrak{i}_E$ is a bijection. We use $\mathfrak{i}_E^{-1}$ to denote the inverse mapping. Moreover, we have that for any $\mathbf{x}, \mathbf{y} \in \mathbb{S}_0^+$ and $\alpha, \beta \in \mathbb{R}$:

$$(\alpha \odot \mathbf{x}) \oplus (\beta \odot \mathbf{y}) = \mathfrak{i}_E^{-1}(\alpha\, \mathfrak{i}_E(\mathbf{x}) + \beta\, \mathfrak{i}_E(\mathbf{y}))$$

and

$$\langle \mathbf{x}, \mathbf{y} \rangle_a = \langle \mathfrak{i}_E(\mathbf{x}), \mathfrak{i}_E(\mathbf{y}) \rangle\,,$$

showing that $\mathfrak{i}_E$ is an isometry.

Even though the isometry defined by the isometric log-ratio transformation has several interesting properties, from a data-analysis point of view, a transformation is mainly useful if the resulting coordinates can be interpreted in a simple manner. To obtain an interpretable coordinate representation, an interpretable basis is needed. Here we give an example (adapted from [5]) that illustrates how an interpretable basis can be chosen. Moreover, this example will turn out to be particularly interesting in Part IV of this dissertation.

As a starting point, given an orthonormal basis $E = \{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$ of $\mathbb{S}_0^d$, let $\mathbf{\Phi}$ be a $d \times (d-1)$ matrix such that the $i$the column of $\mathbf{\Phi}$ equals $\mathfrak{g}(\mathbf{e}_i)^\top$. Interestingly, we have that $\mathbf{\Phi}^\top \mathbf{\Phi} = \mathbf{I}_{d-1}$. It can be shown that [7], for any $\mathbf{x} \in \mathbb{S}_0^+$:

$$\mathfrak{i}_E(\mathbf{x}) = \mathbf{\Phi}^\top \mathfrak{g}(\mathbf{x}) \qquad \text{and} \qquad \mathbf{x} = \mathfrak{c}(\exp(\mathbf{\Phi}\, \mathfrak{i}_E(\mathbf{x})))\,.$$

As the centered log-ratio transformation that is used to define $\mathbf{\Phi}$ is an isometry, we can translate the problem of choosing a basis E into the problem of choosing an appropriate matrix $\mathbf{\Phi}$. Therefore, for notational convenience, we write $\mathfrak{i}_E = \mathfrak{i}_{\mathbf{\Phi}}$. Following the example in Egozcue *et al.* [7], we can use the following procedure to obtain this matrix for $\mathbb{S}_0^d$ (these steps are illustrated in Table 2.5 for $d = 5$).

1. Construct a recursive binary partitioning matrix $\mathbf{B}$, see Appendix 2.A for a clarification on how such a matrix can be constructed and interpreted.

2. Let r_j, (resp. s_j) be the number of times +1 (resp. -1) occurs in the jth column of the binary partitioning matrix $\mathbf{B}$.

3. We now let

$$\mathbf{\Phi}_{i,j} = \begin{cases} 0 & \text{, if } \mathbf{B}_{i,j} = 0\,, \\ \frac{1}{r_j}\sqrt{\frac{r_j s_j}{r_j+s_j}} & \text{, if } \mathbf{B}_{i,j} = +1\,, \\ -\frac{1}{s_j}\sqrt{\frac{r_j s_j}{r_j+s_j}} & \text{, if } \mathbf{B}_{i,j} = -1\,. \end{cases} \tag{2.1}$$

This procedure defines the matrix $\mathbf{\Phi}$ completely. From this matrix, we can derive the corresponding orthonormal basis E of $\mathbb{S}_0^d$. Now, for given matrices $\mathbf{B}$, $\mathbf{\Phi}$ and $\mathbf{x} \in \mathbb{S}_0^d$, let $\mathbf{x}^* = \mathfrak{i}_E(\mathbf{x})$ and let r_j (resp. s_j) be defined as before, we have that

$$\mathbf{x}_j^* = \sqrt{\frac{r_j s_j}{r_j + s_j}} \ln \left[\frac{\left(\prod_{\{i|\mathbf{B}_{i,j}=+1\}} x_i \right)^{1/r_j}}{\left(\prod_{\{i|\mathbf{B}_{i,j}=-1\}} x_i \right)^{1/s_j}} \right]. \tag{2.2}$$

This coordinate has a simple interpretation. Considering the partitioning encoded in the jth column of $\mathbf{B}$, the sign of $\mathbf{x}_j^*$ expresses whether the geometric mean of the elements belonging to the positive class (+1) is greater than the geometric of the elements belonging to the negative class (-1). Moreover, $|\mathbf{x}_j^*|$ is proportional to the difference between these geometric means. The coefficient $\sqrt{\frac{r_j s_j}{r_j+s_j}}$ ensures that different coordinates are scaled appropriately. Due to this scaling, the sizes $|\mathbf{x}_l^*|$ and $|\mathbf{x}_l^*|$ of two coordinates can be compared in a meaningful manner.

Table 2.5: (a) The central part of this table represents a 5×4 partitioning matrix $\mathbf{B}$. The bottom rows count the number of times the labels +1 (represented by r) and -1 (represented by s) appear in each column, (b) The central part of this table visualizes the matrix $\mathbf{\Phi}$ that is computed based on $\mathbf{B}$, using Eqn. (2.1).

Order	1	2	3	4
x_1	+1	0	0	+1
x_2	+1	0	0	-1
x_3	-1	+1	+1	0
x_4	-1	+1	-1	0
x_5	-1	-1	0	0
r	2	2	1	1
s	3	1	1	1

(a)

Order	1	2	3	4
x_1	$\sqrt{\frac{3}{10}}$	0	0	$\sqrt{\frac{1}{2}}$
x_2	$\sqrt{\frac{3}{10}}$	0	0	$-\sqrt{\frac{1}{2}}$
x_3	$-\sqrt{\frac{2}{15}}$	$\sqrt{\frac{1}{6}}$	$\sqrt{\frac{1}{2}}$	0
x_4	$-\sqrt{\frac{2}{15}}$	$\sqrt{\frac{1}{6}}$	$-\sqrt{\frac{1}{2}}$	0
x_5	$-\sqrt{\frac{2}{15}}$	$-\sqrt{\frac{2}{3}}$	0	0

(b)

2.5. Probability distributions on the simplex

The Dirichlet distribution remains the most popular distribution on the simplex (even though other distributions exist, see for instance [8]). When a random vector $\mathcal{X} = (\mathcal{X}_1, \ldots, \mathcal{X}_d)$ is Dirichlet distributed with parameter vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)^\top \in \mathbb{R}_0^{+,d}$, its probability density function $\rho_{\mathcal{X}}$ is given by

$$\rho_{\mathcal{X}}(\mathbf{x}; \boldsymbol{\beta}) = \frac{1}{B(\boldsymbol{\beta})} \prod_{k=1}^{d} x_k^{\beta_k - 1} ,$$

for $\mathbf{x} \in \mathbb{S}_0^d$. $B(\boldsymbol{\beta})$ is a constant calculated as

$$B(\boldsymbol{\beta}) = \frac{\prod_{k=1}^{d} \Gamma(\beta_k)}{\Gamma\left(\sum_{k=1}^{d} \beta_k\right)} ,$$

and $\Gamma(.)$ denotes the Gamma function. Let us define s as

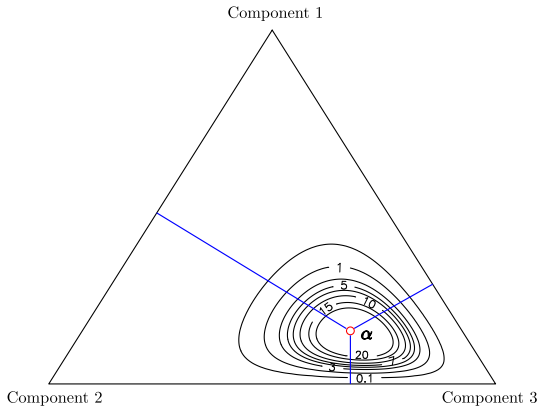
$$s = \sum_{k=1}^{d} \beta_k ,$$

then the Dirichlet distribution has an expected value of $(\mathbb{E}[\mathcal{X}_1], \ldots, \mathbb{E}[\mathcal{X}_d]) = (\beta_1/s, \ldots, \beta_d/s)$.

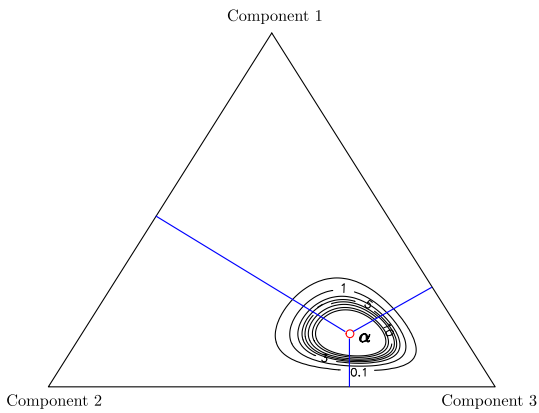
s is called the concentration parameter of the distribution. The concentration parameter allows an intuitive alternative parametrization of the Dirichlet distribution. Thereto, let $\boldsymbol{\alpha} = \boldsymbol{\beta}/s$. Naturally, we have that $\boldsymbol{\alpha} \in \mathbb{S}_0^d$ and $\rho_{\mathcal{X}}(\mathbf{x}; \boldsymbol{\beta}) = \rho_{\mathcal{X}}(\mathbf{x}; s\,\boldsymbol{\alpha})$. Figure 2.1 shows two contour plots of the Dirichlet distribution on $\mathbb{S}_0^3$. In both panels, the parameter vector $\boldsymbol{\alpha}$ is the same but the concentration parameter s is different in both panels.

2.6. Compositional data analysis in this dissertation

In this chapter, a brief overview of the field of compositional data analysis was presented. Throughout this dissertation, we will often rely on some of the principles that have been described in this chapter. Nevertheless, as stated at the beginning of this chapter, it is not our aim to contribute to the fundamentals of compositional data analysis. Instead, we will use the mathematical tools that have been developed in this field as well as the underlying philosophy repeatedly within this dissertation.



(a)



(b)

Figure 2.1: Contour plot of the probability distribution function of a Dirichlet distributed random vector with $\boldsymbol{\alpha} = (0.15, 0.25, 0.60)^\top$. In panel (a) $s = 40$ and in panel (b) $s = 100$.

The attentive reader may have noticed that the fundamentals of compositional data analysis are in conflict with a part of his own opinion. Indeed, the strict focus on log-ratios within this field may lead to problems on several occasions. Most importantly, the use of log-ratios hampers the analysis of data that contain zeros. Most of the analysis methods that have been presented here break down when zeros are involved. Secondly, the methodology that is put forward in this chapter implies that observational noise has a multiplicative effect on the compositions. However, in particular when this noise is due to measuring equipment or rounding, this assumption may be questionable in some cases. This problem will occur in Part IV of this dissertation (where the Aitchison distance is used on several occasions). Interestingly, for most datasets we have seen, a visual inspection of the log-ratio transformed data reveals inexplicable patterns at the boundaries of the sample space. This phenomenon will appear in the experimental results in Part IV of this dissertation. Moreover, in Part III of this dissertation, we will be using the linear mixture model (LMM), which is a very natural model to consider for describing mixtures. Unfortunately, there seems to be no clear link between the LMM and the philosophy that is set forth in this chapter.

2.A. The recursive binary partitioning matrix

A recursive binary partitioning matrix can be seen as a matrix that encodes a recursive binary partitioning of a discrete set. A recursive binary partitioning can most easily be represented by a binary tree. For example, Figure 2.2 visualizes a recursive binary partitioning of a set of 5 components by means of a tree. Naturally, for any partitioning tree (of d components) we have that the number of terminal nodes equals d. Each split in this tree corresponds to a column of the binary partitioning matrix $\mathbf{B} \in \{-1, 0, 1\}^{5\times 4}$. The tree in this figure leads to the following matrix:

$$\mathbf{B} = \begin{pmatrix} +1 & 0 & 0 & +1 \\ +1 & 0 & 0 & -1 \\ -1 & +1 & +1 & 0 \\ -1 & +1 & -1 & 0 \\ -1 & -1 & 0 & 0 \end{pmatrix}.$$

For example, the second column represents the split of the components 3, 4 and 5. As components 1 and 2 are not involved in the split, we have that $\mathbf{B}_{1,2} = \mathbf{B}_{2,2} = 0$. Moreover, in this split, the 5th component is separated from the 3rd and the 4th component. Consequently, have that $\mathbf{B}_{3,2} = \mathbf{B}_{4,2} = +1$ and on the other hand $\mathbf{B}_{5,2} = -1$ (notably, +1 and -1 could have been interchanged).

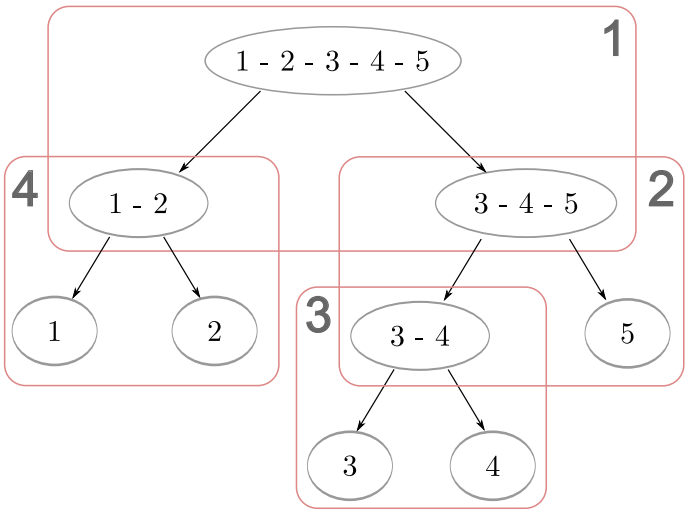


Figure 2.2: Visualization of a recursive binary partitioning by means of a tree for $d = 5$ components. Each split corresponds to a column in the $d \times (d-1)$ matrix $\mathbf{B}$.

3 Mathematical optimization

Mathematical optimization is a central theme in this dissertation. This chapter outlines the general concept of mathematical optimization. The aim of this chapter is threefold. Firstly, it serves as a general introduction to the field of mathematical optimization and can therefore be used to position the contributions of this dissertation in the broad field of mathematical optimization. Secondly, it collects several definitions and properties that will be used later in this work. Thirdly, it may assist the reader who is less familiar with mathematical optimization in appreciating some of the (technical) results in this dissertation.

Most of the material that is presented in this chapter can be found in the textbooks *Convex Optimization* [10] (a standard work on modern convex optimization) and *Numerical Optimization* [11] (a standard work on nonlinear optimization). To a smaller extent, the textbook *Global Optimization: Deterministic Approaches* [12] was used as reference on branch and bound methods. The textbook *Combinatorial Optimization: Algorithms and Complexity* [13] was used as a reference on combinatorial optimization and time complexity.

This chapter is organized as follows:

- In Section 3.1, we illustrate the manner in which mathematical optimization will be used in this dissertation by means of an example.
- In Section 3.2, several definitions and conventions are presented that will be used throughout this dissertation.
- In Section 3.3, several classes of optimization problems are discussed.
- In Section 3.4, the class of continuous optimization problems is briefly reviewed. Several known results on continuous optimization (that will be used in this dissertation) are summarized. Moreover, some popular solution strategies and software implementations are briefly reviewed. This section is mainly intended for the inexperienced reader and can safely be skipped by the reader who is familiar with continuous optimization.
- In Section 3.5, we briefly describe the class of discrete optimization problems (can be skipped safely by the experienced reader).

3.1. The essence of optimization

Numerous problems in engineering, statistics, physics, ..., but also in everyday life involve the search for an optimal state. Typically such questions are phrased in

loose wordings (using a natural language). As an example, consider the following problem:

" Given the dataset in Table 3.1, define a line that optimally separates the points in the positive class from the points in the negative class."

The attentive reader may have several remarks regarding the way this problem is described (generally, we say that this problem is ill-defined). Indeed, one could for example wonder what is meant by an *optimal separation of points*. Nevertheless, problem settings as the one above are rather common in several fields of applied research. In the following paragraph, we elaborate on this particular example and outline a general strategy that can be followed for solving this and similar problems. Even though the problem setting in this example is not new[1], the general strategy for solving it is very illustrative for the general philosophy of this dissertation.

1. As a first step, let us recall that the object we are looking for is a line. Mathematically, given three scalars a_1, a_2 and b, a line is represented by the set $\{(x_1, x_2) \mid a_1x_1 + a_2x_2 + b = 0\}$. In essence, this trivial observation can be used to translate the problem of finding a line into the problem of finding three scalars a_1, a_2 and b. We call a_1, a_2 and b the optimization variables.

2. Secondly, the optimal line should separate the points of the two classes. In essence, this requirement puts constraints on the values that the optimization variables can take. Mathematically, this requirement can be translated into a set of inequalities. For notational purposes, let $\mathbf{D}^+$ (resp. $\mathbf{D}^-$) be a dataset that contains the coordinates of points belonging to the positive (resp. negative) class ($\mathbf{D}^+_{i,j}$ represents the jth coordinate of the ith point in this dataset). Using this notation, the following inequalities are obtained:

$$a_1\mathbf{D}^+_{i,1} + a_2\mathbf{D}^+_{i,2} + b \geq 0\,, \qquad i = 1, \ldots, 4\,, \tag{3.1}$$

$$a_1\mathbf{D}^-_{i,1} + a_2\mathbf{D}^-_{i,2} + b \leq 0\,, \qquad i = 1, \ldots, 4\,, \tag{3.2}$$

This set of eight inequalities puts constraints on the optimization variables encoding the requirement that the optimal line should separate the classes.

[1] The way this problem is solved is inspired by the general philosophy that forms the basis of support vector machines [14].

Table 3.1: Artificial dataset with two features X_1 and X_2 and label Y.

X_1	X_2	Y	X_1	X_2	Y
0.41	0.18	+	0.18	0.42	-
0.62	0.20	+	0.21	0.80	-
0.85	0.31	+	0.43	0.57	-
0.64	0.41	+	0.72	0.81	-

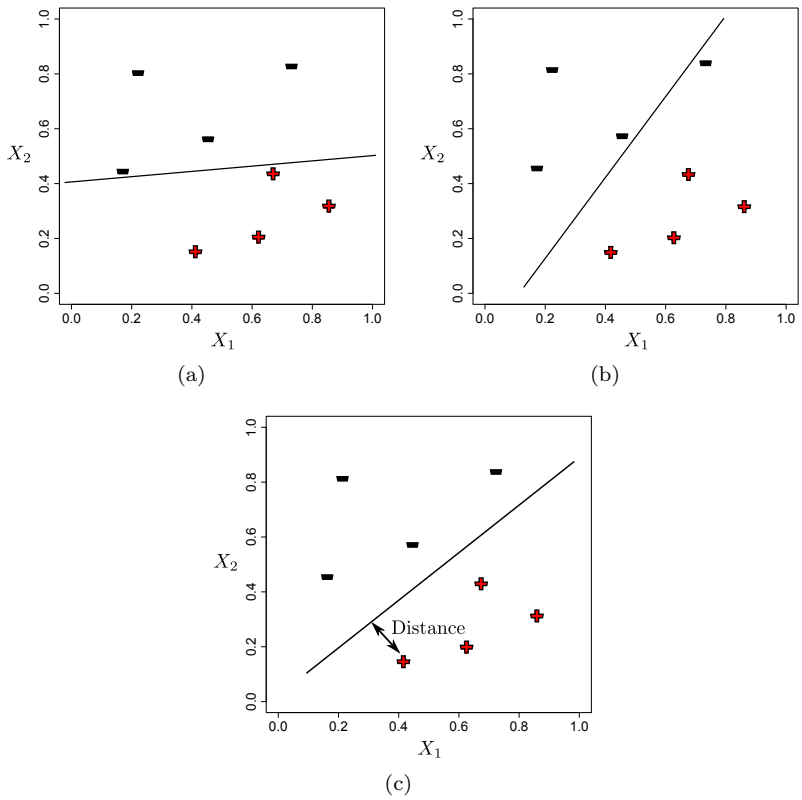


Figure 3.1: Visualization of the dataset in Table 3.1 (negative class (black) and positive class (red)) and three potential (separating) lines. Panels (a) and (c) represent feasible solutions, panel (b) represents an infeasible solution.

These constraints are called problem constraints and the set of the triples (a_1, a_2, b) that respect these constraints is called the feasible set. Figure 3.1 shows three situations, panels (a) and (c) represent feasible lines, panel (b) represents an infeasible line.

3. Thirdly, the line should *optimally* separate the classes. Unfortunately, the problem description does not define an optimality criterion. However, when looking at the plots in Figure 3.1, it would be natural to prefer the line in panel (c) over the line in panel (a). Nevertheless, the problem description does not allow us to express such a preference without making additional assumptions. As a result, in order to proceed, assumptions need to be made regarding what is meant by *optimal*. Such assumptions can be expressed by means of a score function $f : \mathbb{R}^3 \to \mathbb{R}$, that assigns a score to each feasible triple (a_1, a_2, b). Subsequently, we can attempt to find the triple that maximizes this score and use this triple to construct the separating line. As

a score function, the Euclidean distance from the line to the nearest point can be used. Mathematically, this score function is described as follows:

$$f(a_1, a_2, b) = \min_i \left(\frac{|a_1 \mathbf{D}_{i,1} + a_2 \mathbf{D}_{i,2} + b|}{\sqrt{a_1^2 + a_2^2}} \right),$$

where $\mathbf{D}$ is the dataset that contains the coordinates of all points.

4. The score function f and the inequality constraints defined in step 3 can now be used to define the following formal optimization problem: "*Find the triple* (a_1, a_2, b) *that maximizes* f *and respects constraints (3.1) and (3.2)*." This problem can be denoted as follows:

$$\begin{aligned} \underset{(a_1,a_2,b)\in\mathbb{R}^3}{\text{maximize}} \quad & \min_i \left(\frac{|a_1 \mathbf{D}_{i,1} + a_2 \mathbf{D}_{i,2} + b|}{\sqrt{a_1^2 + a_2^2}} \right) \\ \text{subject to} \quad & a_1 \mathbf{D}^+_{i,1} + a_2 \mathbf{D}^+_{i,2} + b \geq 0\,, \qquad i = 1, \ldots, 4\,, \\ & a_1 \mathbf{D}^-_{i,1} + a_2 \mathbf{D}^-_{i,2} + b \leq 0\,, \qquad i = 1, \ldots, 4\,. \end{aligned}$$

Unfortunately, even though it can be proven that this optimization has at least one solution, most standard numerical optimization solvers will not be able to solve this problem directly. However, the following optimization problem is equivalent (two optimization problems are equivalent if the solution of the first can readily be obtained from the solution of the second, and vice verse) to the original problem, but this new problem can be solved easily:

$$\begin{aligned} \underset{(a_1,a_2,b,t)\in\mathbb{R}^4}{\text{maximize}} \quad & t \\ \text{subject to} \quad & a_1 \mathbf{D}^+_{i,1} + a_2 \mathbf{D}^+_{i,2} + b \geq t\,, \qquad i = 1, \ldots, 4\,, \\ & a_1 \mathbf{D}^-_{i,1} + a_2 \mathbf{D}^-_{i,2} + b \leq -t\,, \quad i = 1, \ldots, 4\,, \\ & a_1^2 + a_2^2 \leq 1\,. \end{aligned}$$

We will return on this issue later.

5. As a final step, the optimization problem needs to be solved. In general, three options exist: (1) solve the problem by hand, (2) implement a procedure that is capable of solving this problem, or (3) use existing numerical solvers to solve this problem. Because of the size of the problem, solving the problem by hand is highly unpractical. Moreover, even though several numerical procedures have been described in literature that can solve problems like the one above, making an efficient implementation of these procedures is a highly specialized task, requiring years of experience. Because of that, we will often be using existing high-quality implementations of these procedures.

The five steps described above illustrate the translation of a loosely formulated problem to a well-defined optimization problem. These steps are at the center of several parts of this dissertation. Interestingly, when looking at the definition of mathematical optimization provided by INFORMS[2], it can be seen that this definition perfectly covers these five steps. This contrasts the common perception (see for instance on Wikipedia) of mathematical optimization as a field that solely focuses on the development and application of numerical procedures for solving optimization problems.

3.2. Mathematical optimization: definitions and conventions

3.2.1. The (mathematical) optimization problem

As a starting point of this section, we define a mathematical optimization problem. This definition is commonly used in textbooks such as [11] and [10].

Definition 3.1 ((mathematical) Optimization problem)**.** *Consider a set $X \subseteq \mathbb{R}^n$ and the (vector) functions $f : X \to \mathbb{R}$, $\mathbf{g} : X \to \mathbb{R}^p$ and $\mathbf{h} : X \to \mathbb{R}^q$. A (mathematical) optimization problem is the problem of finding an element $\mathbf{x} \in X$ that minimizes the function f and respects the vector inequality constraint $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p$ and the vector equality constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}_q$. We denote such a problem as:*

$$\begin{aligned} \underset{\mathbf{x} \in X}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p \,, \\ & \mathbf{h}(\mathbf{x}) = \mathbf{0}_q \,. \end{aligned}$$

The set X is called the *domain* of the optimization problem. The vector $\mathbf{x} = (x_1, \ldots, x_n)^\top$ is called the *optimization variable*, f is the *objective function*, $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p$ is a *vector inequality constraint* and $\mathbf{h}(\mathbf{x}) = \mathbf{0}_q$ is a *vector equality constraint*. A point $\mathbf{x} \in X$ is said to be feasible if it respects the vector inequality constraint $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p$ and vector equality constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}_q$. Moreover, the set of all feasible points is called the *feasible set*. When the feasible set is empty, we say that the problem is *unfeasible*.

The *optimal value* p^* of an optimization problem is:

$$p^* = \inf\{f(\mathbf{x}) \mid \mathbf{x} \in X \,, \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p \,, \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\}$$

[2] the society for professionals in the field of operations research (O.R.), management science, and analytics

We allow that $p^* = \pm\infty$. More precisely, when there exists a sequence of feasible points such that $f(\mathbf{x}_k) \to -\infty$ as $k \to +\infty$, we say that the problem is *unbounded below*.

Moreover, we call a point $\mathbf{x}^*$ an *optimal point* if it is feasible and $f(\mathbf{x}^*) = p^*$ (or equivalently, $\mathbf{x}^*$ solves the optimization problem). The set of all optimal points is called the *optimal set*.

Now, let $X = \mathbb{R}^n$, we say that $\mathbf{x}^\bullet$ is a *locally optimal point* if there exists an $r > 0$ such that:

$$f(\mathbf{x}^\bullet) = \inf\{f(\mathbf{x}) \mid \mathbf{x} \in X\,, \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p\,, \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\,, \|\mathbf{x} - \mathbf{x}^\bullet\|_2 \leq r\}\,.$$

Additionally, we call a locally optimal point $\mathbf{x}^\bullet$ a *strict locally optimal point* if we can find an $r > 0$ such that the following implication holds:

$$(\mathbf{x} \in \{\mathbf{x} \in X \mid \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p\,, \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\,, \|\mathbf{x} - \mathbf{x}^\bullet\|_2 \leq r\} \wedge f(\mathbf{x}) = f(\mathbf{x}^\bullet)) \Rightarrow \mathbf{x} = \mathbf{x}^\bullet\,.$$

Note that the definition of a locally optimal point can be extended to any metric space X. Here, we only consider the case where the metric $d(\cdot,\cdot)$ is given by $d(\mathbf{x}_1, \mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|_2$.

Note 1: Within this dissertation, a *solution* of an optimization problem is not necessarily an optimal point. Instead, we loosely say that a point $\mathbf{x}$ is a solution of an optimization problem if it is a feasible point that results from an attempt to solve that optimization problem.

Note 2: Definition 3.1 defines an optimization problem as a minimization problem. However, in this dissertation we will encounter both minimization problems and *maximization problems*. A maximization problem is defined as follows. Given the functions $f : X \to \mathbb{R}$, $\mathbf{g} : X \to \mathbb{R}^p$ and $\mathbf{h} : X \to \mathbb{R}^q$. A maximization problem is the problem of finding an element $\mathbf{x} \in X$ that maximizes the function f and respects the vector inequality constraint $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p$ and the vector equality constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}_q$. We denote such a problem as:

$$\begin{aligned} \underset{\mathbf{x} \in X}{\text{maximize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p\,, \\ & \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\,. \end{aligned}$$

An optimal point of this problem is a feasible point $\mathbf{x}$ for which

$$f(\mathbf{x}) = \sup\{f(\mathbf{x}) \mid \mathbf{x} \in X\,, \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p\,, \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\}\,.$$

It is well known that the following statements are equivalent:

- $\mathbf{x}$ is an optimal point of the maximization problem with objective function f

and constraint functions $\mathbf{g}$ and $\mathbf{h}$.

- $\mathbf{x}$ is an optimal point of the minimization problem with objective function $-f$ and constraint functions $\mathbf{g}$ and $\mathbf{h}$.

Due to this equivalence, we can transform any maximization problem into a minimization problem. **Note 3**: Occasionally, we use the term *mathematical program* to refer to an optimization problem. These terms can be used interchangeably.

An instance of an optimization problem

As mentioned above, an optimization problem is completely defined by its objective function f, its inequality constraint function $\mathbf{g}$ and its equality constraint function $\mathbf{h}$. However, when speaking of an optimization problem, often a distinction is made between a general optimization problem, with generic functions f, $\mathbf{g}$ and $\mathbf{h}$ and an *instance of an optimization problem*, in which these functions are precisely specified. We will be making this distinction several times throughout this dissertation.

3.2.2. General definitions

This section collects several definitions and properties that are frequently used within the field of mathematical optimization. Most of these definitions will be used on multiple occasions within this dissertation. We will also state the relevance of (most) definitions w.r.t. the scope of this dissertation. Most of the definitions presented here can be found in textbooks on convex analysis such as [15].

Sets

Whenever a set S is a subset of the n-dimensional Euclidean space, we write $S \subseteq \mathbb{R}^n$.

We start with the definition of a convex set. As we will see later, convex sets play a dominant role in the field of mathematical optimization. Moreover, convex sets will be used extensively in Part III of this dissertation.

Definition 3.2 (Convex set)**.** *A set $S \subseteq \mathbb{R}^n$ is convex if for any pair of points $\mathbf{x}_1$, $\mathbf{x}_2 \in S$ and scalar $\theta \in [0, 1]$, we have that*

$$\theta\, \mathbf{x}_1 + (1 - \theta)\, \mathbf{x}_2 \in S\,.$$

As a special type of convex sets, we define convex polytopes (which are n-dimensional generalizations of convex polyhedrons).

Definition 3.3 (Convex polytope)**.** *A set $S \subseteq \mathbb{R}^n$ is called a convex polytope if there exists a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ and a vector $\mathbf{b} \in \mathbb{R}^m$ such that we have the*

following identity:

$$S = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} + \mathbf{b} \leq \mathbf{0}_m\}\,.$$

As can be expected from its name, a convex polytope is a convex set. The proof is trivial.

Definition 3.4 (Cone)**.** *A set $S \subseteq \mathbb{R}^n$ is called a cone if for any $\mathbf{x} \in S$ and $\theta \in [0, +\infty[$, we have that:*

$$\theta\,\mathbf{x} \in S\,.$$

One can easily prove that a cone is not necessarily convex. However, the class of convex cones is very important in modern mathematical optimization. More precisely, we will mainly be interested in proper cones.

Definition 3.5 (Proper cone)**.** *A set $S \subseteq \mathbb{R}^n$ is called a proper cone if*

- *S is a convex cone,*
- *S is closed,*
- $\mathbf{int}(S) \neq \emptyset$[3]*,*
- *S contains no lines (we say S is pointed).*

Definition 3.6 (Dual cone)**.** *For a given proper cone $K \subset \mathbb{R}^n$, the set*

$$K^* = \{\mathbf{y} \in \mathbb{R}^n \mid (\forall\,\mathbf{x} \in K)(\mathbf{x}^\top\,\mathbf{y} \geq 0)\}\,,$$

is the dual cone of K.

Definition 3.7 (L_2 Lorentz cone)**.** *The set*

$$K_{L_2} = \{(\mathbf{x}, t) \in \mathbb{R}^{n+1} \mid \|\mathbf{x}\|_2 \leq t\}\,,$$

is the L_2 Lorentz cone.

Definition 3.8 (Sublevel set)**.** *For a given function $f : X \to \mathbb{R}$ and a scalar $a \in \mathbb{R}$ the set $\{\mathbf{x} \in X \mid f(\mathbf{x}) \leq a\}$ is called the a-sublevel set of f.*

Definition 3.9 (Convex hull)**.** *Let $S \subset \mathbb{R}^n$, the convex hull* $\mathrm{conv}(S)$ *of S is the following set*

$$\mathrm{conv}(S) = \left\{ \sum_{i=1}^{k} \theta_i\,\mathbf{a}_i \mid \mathbf{a}_i \in S,\, \theta_i \geq 0,\, i = 1, \ldots, k\,;\, \sum_{i=1}^{k} \theta_i = 1 \right\}\,.$$

Definition 3.10 (Supporting hyperplane)**.** *Let $S \subset \mathbb{R}^n$ and $\mathbf{x}_0 \in \mathbf{bnd}(S)$. If $\mathbf{a} \in \mathbb{R}^n$ (and $\mathbf{a} \neq \mathbf{0}_n$) such that*

[3] The boundary of a set S (denoted $\mathbf{bnd}(S)$) is the subset of S that contains the points which can be approached both from S and from the outside of S. The interior of the set is $\mathbf{int}(S) = S \setminus \mathbf{bnd}(S)$

- $\mathbf{a}^\top \mathbf{x} \leq \mathbf{a}^\top \mathbf{x}_0$ *for all* $\mathbf{x} \in S$*, or*
- $\mathbf{a}^\top \mathbf{x} \geq \mathbf{a}^\top \mathbf{x}_0$ *for all* $\mathbf{x} \in S$*,*

then the hyperplane $\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{a}^\top \mathbf{x} = \mathbf{a}^\top \mathbf{x}_0\}$ *is called a supporting hyperplane of* S *at* $\mathbf{x}_0$*.*

(Quasi-)Convex functions

Just as convex sets, convex functions play a dominant role in the field of mathematical optimization. In the following definitions, we will assume that X is a convex subset of $\mathbb{R}^n$.

Definition 3.11 (Convex function)**.** *Let* X *be a convex subset of* $\mathbb{R}^n$*. A function* $f : X \to \mathbb{R}$ *is called convex if for any* $\mathbf{x}_1$*,* $\mathbf{x}_2 \in X$ *and* $\theta \in [0, 1]$ *we have that*

$$f(\theta\, \mathbf{x}_1 + (1 - \theta)\, \mathbf{x}_2) \leq \theta\, f(\mathbf{x}_1) + (1 - \theta)\, f(\mathbf{x}_2)\,.$$

When a function $f : X \to \mathbb{R}$ is twice differentiable over its domain and $\nabla_{\mathbf{xx}} f(\mathbf{x})$ is positive semi-definite for every $\mathbf{x} \in X$, we have that f is convex.

Definition 3.12 (Concave function)**.** *Let* X *be a convex subset of* $\mathbb{R}^n$*. A function* $f : X \to \mathbb{R}$ *is called concave if* $-f$ *is convex.*

Even though convex functions are amenable for optimization purposes, requiring convexity is sometimes too restrictive. Instead, the class of quasi-convex functions is more general and has some properties that allow for efficient optimization as well.

Definition 3.13 (Quasi-convex function)**.** *Let* X *be a convex subset of* $\mathbb{R}^n$*. A function* $f : X \to \mathbb{R}$ *is called quasi-convex if all its sublevel sets are convex.*

Definition 3.14 (Quasi-concave function)**.** *Let* X *be a convex subset of* $\mathbb{R}^n$*. A function* $f : X \to \mathbb{R}$ *is called quasi-concave if* $-f$ *is quasi-convex.*

It can easily be shown that each convex function is quasi-convex. However, the converse is not generally true.

Generalized inequalities

Traditionally, given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we write $\mathbf{x} \leq \mathbf{y}$ if and only if $x_i \leq y_i$ for each $i = 1, \ldots, n$. Because of that, we have the following trivial equivalence:

$$\mathbf{x} \leq \mathbf{y} \iff \mathbf{y} - \mathbf{x} \in \mathbb{R}^n_+\,.$$

It is well known that this vector inequality (as it operates on vectors) defines a partial ordering on $\mathbb{R}^n$. A *generalized inequality* defines a partial ordering that

has many of the properties of the vector inequality above. To define a generalized inequality, we will be using proper cones (this presentation is based on [10]).

Using a proper cone $K \subset \mathbb{R}^n$, a partial ordering (or a generalized inequality) is defined as follows

$$\mathbf{x} \preccurlyeq_K \mathbf{y} \quad \text{if} \quad \mathbf{y} - \mathbf{x} \in K\,.$$

Equivalently, a strict partial ordering is defined as follows

$$\mathbf{x} \prec_K \mathbf{y} \quad \text{if} \quad \mathbf{y} - \mathbf{x} \in \mathbf{int}(K)\,.$$

Generalized inequalities can be used to define (convex) sets. As an example, consider a proper cone K, a $m \times n$ matrix $\mathbf{B}$ and an n-vector $\mathbf{b}$. It can easily be shown that

$$B = \{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \preccurlyeq_K \mathbf{b}\}$$

is a convex set.

3.3. Classes of optimization problems

Once an optimization problem has been constructed, the next logical step is to try to solve it (*i.e.* find an optimal point $\mathbf{x}^*$). Unfortunately, due to the general nature of the optimization problem in Definition 3.1, there exists no algorithm that is capable of solving all optimization problems (at least not within a reasonable amount of time). As a result, to be able to study optimization problems and develop new solvers, optimization problems are subdivided into several classes.

A class of optimization problems is a set of instances (of optimization problems) that share some properties. For instance, the class of linear programs[4] is the set of instances for which f, $\mathbf{g}$ and $\mathbf{h}$ are affine vector functions (and $X = \mathbb{R}^n$).

3.3.1. Continuous versus discrete optimization

Optimization problems can (at first sight) be naturally divided in two classes: continuous optimization problems and discrete optimization problems.

In the former class, the optimization variables take real values. Generally, the domain X of the optimization problem is an uncountably infinite set. For all continuous optimization problems within this dissertation, we will assume that $X = \mathbb{R}^n$. Whenever a situation requires that $X \subset \mathbb{R}^n$, this will be encoded by adding inequalities to reduce the feasible set. Typical examples of continuous

[4] We could also say linear optimization problem, but the term linear program is more commonly used.

optimization include linear programming and least squares problems. We will encounter numerous optimization problems that belong to this class throughout this dissertation (mainly in Parts III and IV).

In the latter class, the optimization set is a finite set (or possibly countably infinite). Typical examples include timetabling problems, the traveling salesman problem and the shortest path problem. Within this dissertation, we will be dealing with discrete optimization problems in Part II only.

Typically, the strategies that are used to solve problems from the class of continuous optimization problems differ quite strongly from the strategies that are used to solve discrete optimization problems. It should be noted, however, that there exists a rather smooth transition from the class of discrete problems to the class of continuous optimization problems. For example, in mixed integer linear programs, some of the optimization variables are continuous whereas others are discrete.

3.3.2. Easy to solve versus hard to solve

Mostly, when referring to the difficulty of an optimization problem, we look at the time that it takes to solve the problem (*i.e.* the time it takes to guarantee that an optimal point has been found).

The theoretical framework of time complexity (see for instance [13]) provides a criterion that can be used to distinguish easy from hard problems. Loosely speaking, the time complexity of a problem combined with an algorithm is a function that expresses the time it takes to solve a problem as a function of the size of the problem with that algorithm. As an example, consider the traveling salesman problem. This problem consists of finding the shortest path in a completely connected weighted graph that visits every node exactly once. The size of this problem is measured by the number of nodes n in the graph. Assume that our algorithm exhaustively searches through the space of possible paths. This means that in total $n!$ paths need to be computed. The time complexity of this algorithm thus is $\mathcal{O}(n!)$ (of the order n factorial). It is generally assumed that when the best known algorithm has a complexity that is exponential, the problem is considered hard to solve[5].

Interestingly, with respect to time complexity, it can be noted that the class of linear programs (shortly described before) has been a class of hard optimization problems for quite some time. That is, it is only since the late 1970s that an algorithm exists that can be proven to solve all instances from the class of linear programs in polynomial time. Nevertheless, even today, the simplex algorithm (developed in the late 1940s) is still used even though it has a complexity that is exponential. Despite of these theoretical issues, the simplex algorithm can solve linear programs with

[5] This discussion might insinuate a reference to the class of NP-complete problems, however, this (rather technical) debate is beyond the scope of this dissertation.

hundreds of variables and thousands of constraints within a reasonable amount of time. It even outperforms several polynomial time algorithms.

In this dissertation, even though we take several theoretical aspects regarding time bounds into account, we will mainly be focusing on the practical runtime of an algorithm.

3.4. Continuous optimization problems

In this section, we (briefly) describe the most important classes and elaborate shortly upon the classes that are of special interest for this dissertation.

3.4.1. Smooth versus non-smooth optimization

Within the class of continuous optimization problems, we distinguish between smooth optimization problems and non-smooth optimization problems. A smooth optimization problem is an optimization problem with a smooth objective function, and smooth inequality and equality constraint functions. Here, smooth means that derivatives of these functions with respect to each decision variable (*i.e.* the function gradients), are continuous. If at least one of these functions is not smooth, the problem is non-smooth.

In general, smooth optimization problems can be solved more easily (or at least, the methodology for solving them is more established) than non-smooth optimization problems [16]. However, as we shall see hereafter there exist several broad classes of non-smooth optimization problems that can be solved efficiently as well. Within this dissertation, we will encounter both types of problems.

Smooth optimization: Characterizing locally optimal points

We now consider optimization problems for which the functions $f : X \to \mathbb{R}$, $\mathbf{g} : X \to \mathbb{R}^p$ and $\mathbf{h} : X \to \mathbb{R}^q$ are twice continuously differentiable. In this case there exist several well-known results that can be used to characterize locally optimal points. The most famous of these characterizations are the Karush-Kuhn-Tucker conditions (KKT-conditions) [17]. As these conditions will be used explicitly later in this dissertation (Part III), we briefly elaborate upon them. This presentation is adapted from [11].

Definition 3.15 (Active set)**.** *For a given optimization problem, the active set $\mathcal{A}(\mathbf{x})$ of a point $\mathbf{x} \in X$ is a subset of the equality and inequality constraint functions, including*

- *each function g_i for which $g_i(\mathbf{x}) = 0$,*
- *each function h_i for which $h_i(\mathbf{x}) = 0$.*

We call the elements of $\mathcal{A}(\mathbf{x})$ the active constraints at $\mathbf{x}$.

Definition 3.16 (LICQ)**.** *Given a point $\mathbf{x} \in X$ and its active set $\mathcal{A}(\mathbf{x})$, we say that the linear independence constraint qualification (LICQ) holds if the gradient vectors of the active constraints evaluated at $\mathbf{x}$ are linearly independent.*

Definition 3.17. *The Lagrangian function of an optimization problem is the function*

$$\begin{array}{rccl} \ell: & X \times \mathbb{R}^p \times \mathbb{R}^q & \to & \mathbb{R} \\ & (\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) & \mapsto & f(\mathbf{x}) + \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{x}) + \boldsymbol{\gamma}^\top \mathbf{h}(\mathbf{x}) \,. \end{array}$$

These definitions can now be used to state the KKT conditions, which are first order necessary conditions for locally optimal points.

Proposition 3.1 (KKT conditions)**.** *Suppose that $\mathbf{x}^\bullet$ is a locally optimal point for a given optimization problem with Lagrangian function ℓ and that LICQ holds at $\mathbf{x}^\bullet$, then there exist two Lagrange multiplier vectors $\boldsymbol{\lambda}^\bullet \in \mathbb{R}^p$ and $\boldsymbol{\gamma}^\bullet \in \mathbb{R}^q$ such that*

$$\begin{aligned} \nabla_{\mathbf{x}} \ell(\mathbf{x}^\bullet, \boldsymbol{\lambda}^\bullet, \boldsymbol{\gamma}^\bullet) &= \mathbf{0}_n \,, \\ \mathbf{g}(\mathbf{x}^\bullet) &\leq \mathbf{0}_p \,, \\ \mathbf{h}(\mathbf{x}^\bullet) &= \mathbf{0}_q \,, \\ \boldsymbol{\lambda}^\bullet &\geq \mathbf{0}_p \,, \\ \lambda_i^\bullet \, g_i(\mathbf{x}^\bullet) &= 0 \,, \qquad i = 1, \ldots, p \,. \end{aligned}$$

Unfortunately, the KKT conditions are not sufficient conditions for locally optimal points. However, there exist several classes of optimization problems for which these conditions are sufficient. From a numerical point of view, the KKT conditions are important as well, as most solvers attempt to find a solution to this system of equalities and inequalities. However, as we will be needing sufficient conditions for locally optimal points in Part III, we present the sufficient conditions here as well.

Definition 3.18. *Consider an optimization problem, a given point $\mathbf{x}$ for which the KKT conditions hold with Lagrangian multiplier vectors $\boldsymbol{\lambda}$ and $\boldsymbol{\gamma}$ and the active set $\mathcal{A}(\mathbf{x})$. We define the tangent cone to the feasible set at $\mathbf{x}$ as follows:*

$$\mathbb{F}_1(\mathbf{x}) = \left\{ \alpha \, \mathbf{d} \mid \alpha \in [0, +\infty[\,, \mathbf{d} \in \mathbb{R}^n \,, \begin{array}{ll} \mathbf{d}^\top \nabla_{\mathbf{x}} h_i(\mathbf{x}) = 0 \,, & \text{for } i = 1, \ldots, q \,, \\ \mathbf{d}^\top \nabla_{\mathbf{x}} g_i(\mathbf{x}) = 0 \,, & \text{for all } g_i \in \mathcal{A}(\mathbf{x}) \end{array} \right\}$$

Moreover, the subset of adherent directions of this cone is defined as

$$\mathbb{F}_2(\boldsymbol{\lambda}) = \{ \mathbf{d} \in \mathbb{F}_1 \mid \mathbf{d}^\top \nabla_{\mathbf{x}} g_i(\mathbf{x}) = 0 \text{ for all } g_i \in \mathcal{A}(\mathbf{x}) \text{ with } \lambda_i > 0 \} \,.$$

This definition is used to state the second order necessary conditions.

Proposition 3.2 (Second order necessary conditions)**.** *Suppose that* $\mathbf{x}$ *is a locally optimal point of an optimization problem and that LICQ holds. Let* $\boldsymbol{\lambda}$ *and* $\boldsymbol{\gamma}$ *be two Lagrangian multiplier vectors for which the KKT conditions are satisfied and let* $\mathbb{F}_2(\boldsymbol{\lambda})$ *be defined as above. Then*

$$\mathbf{d}^\top \nabla_{\mathbf{xx}} \ell(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\gamma})\, \mathbf{d} \geq 0\,, \qquad \text{for all } \mathbf{d} \in \mathbb{F}_2(\boldsymbol{\lambda})\,.$$

Similarly, we can state second order sufficient conditions.

Proposition 3.3 (Second order sufficient conditions)**.** *Suppose that for some feasible point* $\mathbf{x}$ *there is a pair of Lagrangian multiplier vectors* $\boldsymbol{\lambda}$ *and* $\boldsymbol{\gamma}$ *such that the KKT conditions are satisfied and let* $\mathbb{F}_2(\boldsymbol{\lambda})$ *be defined as above. If*

$$\mathbf{d}^\top \nabla_{\mathbf{xx}} \ell(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\gamma})\, \mathbf{d} > 0\,, \qquad \text{for all } \mathbf{d} \in \mathbb{F}_2(\boldsymbol{\lambda}), \mathbf{d} \neq 0\,,$$

then $\mathbf{x}$ *is a locally optimal point.*

These propositions provide a useful characterization of locally optimal points. We will be using this characterization in Part III.

Non-smooth optimization: Characterizing locally optimal points

For non-smooth optimization problems, similar characterizations can be made. However, these characterizations are outside the scope of this dissertation.

3.4.2. Convex optimization problems

An optimization problem is called a *convex optimization problem* if f is a convex function, $\mathbf{g}$ is a vector of convex functions and $\mathbf{h}$ is a vector of affine functions. This class is by far the most amenable class for optimization. Indeed, owing to the fact that within this class any locally optimal point is also globally optimal, a procedure that is guaranteed to find a locally optimal point can be used to solve the problem. Moreover, except for some degenerate cases, convex optimization problems have a unique globally optimal point.

Nevertheless, convexity of a problem does not guarantee the existence of an algorithm that allows the problem to be solved in a reasonable amount of time. To be able to provide these guarantees, we need to make further restrictions.

Linear programs

A linear program is an optimization problem with an affine objective function and affine inequality and equality constraint functions. This class of problems is among

the oldest and most extensively studied classes of optimization problems. As a result linear programs are used extensively in a variety of (not to say almost any) research disciplines. In the life sciences several recent examples include optimal diet composition [18], protein structure prediction [19] or the impact of the water and agriculture policy on farming [20]. These examples illustrate that the use of linear programming is very common. A simple search through literature quickly reveals hundreds of relevant publications. Interestingly, even though not stated explicitly, many of the publications that use linear programming in applied domains resemble the flow that was presented in the introduction of this chapter.

We will be using linear programs in Part III of this dissertation.

(convex) Quadratic optimization problems

A quadratic optimization problem is an optimization problem with a quadratic objective function and quadratic inequality and equality constraint functions. It can easily be seen that the class of quadratic optimization problems extends the class of linear programs. However, in general the class of quadratic optimization problems is not convex. Therefore this class is further subdivided into:

- *Linearly constrained convex quadratic optimization problems* [11]: this is the most well-known class of quadratic optimization problems. The objective function is a convex quadratic functions and the inequality and equality constraint functions are linear. As an example, support vector machines [14] lead to this type of optimization problems. Another traditional example is the Markowitz portfolio optimization problem [21]. Linearly constrained convex quadratic optimization problems will be encountered in Part IV.

- *Quadratically constrained convex quadratic optimization problems* [11]: this class extends the previous one by allowing the inequality constraint functions to be convex quadratic functions.

- *Quadratically constrained quadratic optimization problems*: this class is the most general class of quadratic optimization problems and allows (possibly non-convex) quadratic objective functions and quadratic inequality and equality constraint functions. Unfortunately, these problems are generally hard to solve [22].

  As a special case, we consider (generalized) bilinear programs [23]. Bilinear programs are quadratic optimization problems that contain bilinear functions. More precisely, a bilinear function f is a quadratic function that can be written as $f : \mathbb{R}^s \times \mathbb{R}^t \to \mathbb{R}$ such that for a fixed pair $(\mathbf{u}, \mathbf{v}) \in \mathbb{R}^s \times \mathbb{R}^t$, we have that both $f(\mathbf{u}, \cdot)$ and $f(\cdot, \mathbf{v})$ are affine functions. Even though the generalized bilinear program is less known in applied research, it has been used in the life sciences (see for instance [24] for an application in farm

management). We will study a specific case of the bilinear program in Part III of this dissertation.

(convex) Conic optimization problems

The class of conic optimization problems is without doubt the most general class of convex optimization problems (it includes any type of smooth convex quadratic optimization problem [10]). As a general definition, we say that a conic optimization problem is a mathematical optimization problem of the following form.

$$\begin{aligned} \underset{\mathbf{x}\in X}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{b} \preccurlyeq_K \mathbf{0}_p \,, \\ & \mathbf{C}\mathbf{x} + \mathbf{d} = \mathbf{0}_q \,, \end{aligned}$$

where $\mathbf{A} \in \mathbb{R}^{p\times n}$, $\mathbf{b} \in \mathbb{R}^p$, $\mathbf{C} \in \mathbb{R}^{q\times n}$, $\mathbf{d} \in \mathbb{R}^q$ and K is a proper cone. Interestingly, only using a limited number of cones, an overwhelming number of convex mathematical optimization problems can be encoded in a uniform manner. Popular cones include the (second order) Lorentz cone and the semi-definite cone. For instance the second order Lorentz cone allows the inclusion of (non-smooth) constraints of the type

$$\|\mathbf{A}\mathbf{x} + \mathbf{b}\|_2 \leq \mathbf{r}^\top \mathbf{x} + s$$

where $\mathbf{A}$ and $\mathbf{b}$ are defined as before and $\mathbf{r} \in \mathbb{R}^n$, $s \in \mathbb{R}$. This this type of constraints is used extensively in the field of robust optimization (and robust parameter estimation) [25].

In this dissertation, we will use this type of constraints in Parts III and IV.

3.4.3. Quasi-convex optimization problems

A quasi-convex optimization problem is a mathematical optimization problem with a quasi-convex objective function f, a vector $\mathbf{g}$ of convex inequality constraint functions and a vector $\mathbf{h}$ of affine equality constraint functions (this definition is similar to the one given in [26]). In the previous section, we stated that convex optimization problems are amenable for optimization, the most important reason for this being the property that locally optimal points are always globally optimal. Most quasi-convex optimization problems share this property. We say most, because quasi-convex objective functions can have flat regions, and according to the definition of locally optimal points these regions are essentially sets of locally optimal points. This means that (in most cases), here as well, a procedure that guarantees to converge to a locally optimal point can be used to solve the problem.

Unfortunately, a lot of procedures that can efficiently optimize convex optimization problems exploit the convexity of the objective function. This means that these procedures cannot be applied directly to solve quasi-convex optimization problems. However, in several cases, quasi-convex optimization problems can be solved by converting them into a sequence of convex optimization problems [10]. To do this, we define a parametrized family of convex functions (with parameter a) $\phi_a : X^n \to \mathbb{R}$ such that the following implication holds:

$$f(\mathbf{x}) \leq a \Rightarrow \phi_a(\mathbf{x}) \leq 0\,.$$

We can now fix a and solve the following optimization problem.

$$\begin{aligned} \underset{(\mathbf{x},s)\in X\times\mathbb{R}}{\text{minimize}} \quad & s \\ \text{subject to} \quad & \phi_a(\mathbf{x}) \leq s\,, \\ & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p\,, \\ & \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\,. \end{aligned}$$

It is easy to see that when the optimal value (p^*) of the original quasi-convex optimization problem is smaller than or equal to a, the optimal value ($p^\bullet(a)$) of the mathematical optimization problem above will be smaller than or equal to zero. This property can be used to solve the quasi-convex optimization problem:

1. Choose a value for a, and a tolerance ϵ.
2. Compute $p^\bullet(a)$ and go to step 3.
3. **If** $|p^\bullet(a)| \leq \epsilon$: stop, **else**, go to step 4.
4. **If** $p^\bullet(a) < 0$: reduce the value of a and go to step 2, **else**, go to step 5.
5. **If** $p^\bullet(a) > 0$: increase the value of a and go to step 2.

When this procedure is iterated, it will converge to a globally optimal value for the quasi-convex optimization problem.

In this dissertation, we will encounter quasi-convex optimization problems in Part III.

3.4.4. Solving continuous optimization problems

Once an optimization problem has been defined, it generally needs to be solved, *i.e.* an optimal point needs to be found. For some small or trivial cases, finding the optimal point can be done by hand. Unfortunately, most optimization problems cannot be solved by hand. However, in most cases we can use computers to assist us in trying to find an optimal point. Typically, computers use (iterative) numerical procedures to find such optimal points. In fact, the main purpose of the different classes of optimization problems presented before is to allow the development of

specialized and efficient numerical procedures. Indeed, as stated before, there exists no unique numerical procedure that allows to solve all mathematical optimization problems. In fact, a lot of numerical optimization problems cannot be solved at all (even with the help of the most advanced numerical procedures).

Local optimization

The main purpose of using a numerical optimization procedure is to find an optimal point. Unfortunately, finding an optimal point turns out to be a very difficult task in the general case. Finding a locally optimal point is an easier task. Because of that, we will be focusing here on finding locally optimal points. Moreover, as mentioned before, for convex optimization problems, any local optimal point is a globally optimal point.

As the purpose of this dissertation does not aim at the development of novel optimization procedures (at least not for the continuous case), we will not go into much detail here. Instead, we refer the interested reader to several excellent textbooks on continuous numerical optimization (both textbooks [11] and [10] elaborate on the development of algorithms for solving optimization problems). However, as we will be using existing implementations of several numerical procedures, we give a brief overview of the general ideas behind these procedures and some references to existing software.

In the case of smooth continuous optimization, the KKT conditions can be used to characterize locally optimal points. Note that these conditions are simply a system of equalities and inequalities. In essence most numerical optimization procedures exploit this property and will, for a given optimization problem, generate a sequence of points

$$(\mathbf{x}^0, \boldsymbol{\lambda}^0, \boldsymbol{\gamma}^0), \quad (\mathbf{x}^1, \boldsymbol{\lambda}^1, \boldsymbol{\gamma}^1), \quad (\mathbf{x}^2, \boldsymbol{\lambda}^2, \boldsymbol{\gamma}^2), \quad \ldots \quad ,$$

that will (hopefully) converge to a solution of the KKT system. In some cases, the first point of this sequence is provided by the user. During the past 60 years, a variety of methods has been developed that can be used to find such a point. Some of them, such as for instance the famous simplex algorithm by Dantzig, can only be used to solve a specific subclass of optimization problems (linear programs in case of the simplex algorithm), others are more widely applicable. The numerical procedures that are developed to be very generally applicable are often called general purpose solvers.

General purpose solvers

There exists a variety of general purpose solvers. Depending on the philosophy on which they are based or the reference that is used, such methods are called

logarithmic barrier methods, augmented Lagrangian methods, sequential quadratic programming, primal-dual methods or interior point methods (some of the classes mentioned may considered as subclasses of others).

To be useful in practice, a procedure should be implemented in an efficient manner. Indeed, it is generally known that an efficient implementation is key to the usability of any procedure. Making such an efficient implementation is often a highly specialized and time-consuming task. Fortunately, several high-quality implementations of a variety of numerical procedures exist. Such implementations include LANCELOT [27] (an augmented Lagrangian method), IPOPT [28] (a primal-dual interior point solver), or the fmincon-function of MATLAB [29] (containing, amongst others, an implementation of sequential quadratic programming). It should be noted however that, even though these implementations represent the state of the art, they are only local optimization methods. Moreover, when the constraints are non-convex, there is no guarantee that a general purpose solver will even find a feasible point.

Linear and linearly constrained convex quadratic programming solvers

Since the introduction of the first (generally available) numerical procedure for solving linear programs by Dantzig in 1947, there has been a huge interest in the development of efficient numerical procedures for solving linear and linearly constrained convex quadratic optimization problems. These developments have led to a variety of both, freely available and commercial solvers. Popular solvers include the CPLEX solver [30] or the MOSEC solver [31]. Modern implementation implementations that use sparse matrix representations allow problems with tens of thousands of optimization variables and millions of constraints to be solved within several minutes. The main reason for this efficiency is that fact that these methods exploit the specific structure of this class.

Conic programming solvers

In the paragraph above, the efficiency of linear and linearly constrained convex quadratic programming solvers was attributed to the fact that these solvers exploit the problem structure. As the class of conic optimization problems is very large, it can be expected that the problem structure that is shared by all instances in this class is considerably less. In the general case, we could argue that this is indeed the case. However, throughout the past decade the research community of conic programming has mainly been focusing on the development of efficient optimization procedures for a limited number of proper cones (mainly the second order Lorentz cone and the semi-definite cone). This restriction allows to exploit specific properties of these cones, resulting once more in highly efficient procedures. Examples of broadly used (and freely available) implementations are SeDuMi [32] and SDPT3 [33]. Generally speaking, when a convex optimization problem can be translated into a conic optimization problem that uses the Lorentz cone or the semi-definite cone, it can be solved efficiently. As an example, consider the

optimization problem presented in the first section of this chapter (step 4). We could attempt to use a general purpose solver to solve this problem (in this case, this solver would definitely find the optimal point). However, using the Lorentz cone, this problem can be rewritten as a conic optimization problem. Solving the problem in the latter form will be more efficient.

Global optimization

Essentially, the procedures that have been described before are local optimization procedures. This means that these methods look for locally optimal points. Naturally, for convex optimization problems locally optimal points are globally optimal as well. However, this does not generally hold for non-convex optimization problems. Most non-convex optimization problems have multiple locally optimal points. As the general purpose solvers introduced before can only guarantee convergence to a locally optimal point, the solutions that are offered by these procedures cannot generally be assumed to be globally optimal.

As a simple alternative, we could try to compose an exhaustive list of locally optimal solutions (using the general purpose solvers) and simply take the solution with the minimal objective function value as the global solution. Even though this procedure could be applied in some simple cases, it is often very hard (in terms of time complexity) to compose such an exhaustive list. Therefore, this (at first sight) very simple procedure turns out to be rather impractical. Below, we present several alternatives.

Heuristics

As a first alternative, we could settle for a feasible point that has a low objective function value. In that case, we need to define what we mean with a low objective function value. In some cases, the application at hand can indicate whether a given feasible point has an acceptably low objective function value. Alternatively, we could translate low to "*as low as we can given a limited amount of resources*". In this dissertation, we define a heuristic as a procedure that is intended to find a feasible point with a low objective function value of an optimization problem, without the guarantee that this point is optimal.

As a first heuristic, we can choose a (feasible) point and use this point as a starting point for a general purpose solver. This strategy will generally result in a locally optimal point. However, knowing that locally optimal points can still lead to high the objective function values, such a strategy will not necessarily lead to a result that is satisfactory. As a simple extension, multiple (feasible) points can be chosen and used successively as starting points for a general purpose solver. Subsequently, the minimum of the resulting points can be selected. Such an approach is for instance used in [34].

Alternatively, several heuristics have been developed that are inspired on (evolutionary) processes in nature. Examples include: particle swarm optimization [35], genetic algorithms [36] and simplex simulated annealing [37]. These procedures typically start from a (set of) random point(s) and recursively apply a rule base that acts upon these points. By selecting a well-thought rule base, it is hoped that that at least one of these points will converge to a solution with a low objective function value. Unfortunately, there are close to no guarantees regarding the quality of the solution that will be found. We will use similar procedures (for discrete optimization) in Part II of this dissertation.

Branch and bound (B & B) procedures

The discussion above illustrates that solving general non-convex optimization problems can be hard. In this section, we briefly describe the main principle of branch and bound procedures (see [12] for a standard work). The solutions provided by this class of procedures are guaranteed to be ϵ-optimal. This means that the objective function value of the solution that is obtained is at most ϵ higher than the optimal value. Here, ϵ is an arbitrary tolerance parameter. Even though these methods are very appealing, the main drawback is the time complexity, which can be excessive in some cases. Below we present the main philosophy behind this class of methods. We will develop a branch and bound procedure in Part III of this dissertation.

Definition 3.19 (A (convex) lower bound)**.** *Consider the sets $X \subseteq \mathbb{R}^n$ and $C \subseteq X$ and the functions $f : X \to \mathbb{R}$ and $\bar{f}^C : X \to \mathbb{R}$. We say that $\bar{f}^C$ is a lower bound on f over the set C if, for any $\mathbf{x} \in C$, we have that:*

$$\bar{f}^C(\mathbf{x}) \leq f(\mathbf{x}) \,.$$

Moreover, when $\bar{f}^C$ is convex, we call it a convex lower bound on f.

Similarly, we can define a *convex upper bound.*

To be useful in a B & B setting, it is preferable that $\bar{f}^C$ is an element of a family of functions that is parametrized by C. Moreover, the following (natural) properties are required [38]:

(a) For $C' \subset C \subset \mathbb{R}^n$, we have that $\bar{f}^{C'}(\mathbf{x}) \geq \bar{f}^C(\mathbf{x})$ for all $\mathbf{x} \in C'$.

(b) If $C = \{\mathbf{x}\}$, we have that $\bar{f}^C(\mathbf{x}) = f(\mathbf{x})$.

Property (a) ensures that deleting points from a subset does not lead to a decrease of the lower bound. Property (b) ensures that singleton subsets are not unnecessarily loose.

We will assume that there are no equality constraints (this is not a restriction as any equality constraint can be written as a pair of equality constraints). A branch and bound procedure will try to solve an optimization problem by using convex

lower bounds to define relaxed versions of the original optimization problem.

Definition 3.20 (Convex C-relaxation)**.** *Consider an optimization problem with convex domain X, objective function f and vector inequality constraint function $\mathbf{g} : X \to \mathbb{R}^p$ as well as their convex lower bounds $\bar{f}^C$ and $\bar{\mathbf{g}}^C$, where C is a bounded subset of X. We call the optimization problem*

$$\begin{aligned} \underset{\mathbf{x}\in X}{\text{minimize}} \quad & \bar{f}^C(\mathbf{x}) \\ \text{subject to} \quad & \bar{\mathbf{g}}^C(\mathbf{x}) \leq \mathbf{0}_p \,, \\ & \mathbf{x} \in C \,, \end{aligned}$$

a convex C-relaxation of the original optimization problem.

It can easily be seen that the optimal value of the convex C-relaxation will be smaller than or equal to the optimal value of the original problem (when constrained to C). A branch-and-bound procedure will recursively partition the domain of the original optimization problem and solve a sequence of convex relaxations as illustrated below.

1. Construct a partitioning $P = \{C^1, \ldots, C^s\}$ of the bounded domain X and choose a tolerance parameter ϵ.

2. For each $C^i \in P$ (this is called *bounding*):

   - Find a locally optimal point of the following optimization problem:

     $$\begin{aligned} \underset{\mathbf{x}\in X}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p \,, \\ & \mathbf{x} \in C^i \,, \end{aligned}$$

     we call the locally optimal value the *upper bound* of the optimal value on C^i.

   - Find the optimal point of the convex C^i-relaxation of the optimization problem above. We call this optimal value the *lower bound* on C^i.

3. Let $\mathbf{x}^\bullet$ be the locally optimal point with the lowest objective function value found so far.

   - Remove each $C \in P$ that has a lower bound that is greater than $\mathbf{x}^\bullet$ from P.

   - Remove each $C \in P$ for which the gap between the upper and the lower bound is smaller than ϵ from P.

4. **If** $P \neq \emptyset$: select (by some criterion, called the *branching rule*) an element $C \in P$, partition C and add this partitioning to P, subsequently, delete C

from P and go back to step 2. **Else**: stop, conclude that $\mathbf{x}^\bullet$ is an ϵ-optimal point.

Note that the scheme above is only one variant of the general branch and bound framework. Other variants or practical implementations can slightly deviate from this scheme.

3.5. Discrete optimization problems

The domain of a discrete optimization problem is a finite or countably infinite set [13]. Similar to continuous optimization problems, there exist several classes of discrete optimization problems. Historically, there has always been a strong link between discrete optimization and graph theory. The connection between those two disciplines is very natural as most discrete optimization problems can be seen as graph problems. Because of that, several problem classes have names that are related to graphs, for example: the shortest path problem or the minimum spanning tree problem. Other classes have more problem-oriented names, for example: the traveling salesman problem and vehicle routing problems. However, as we will not be using these classes within this dissertation, we do not elaborate on this topic here. Instead, we will only briefly introduce the class of subset selection problems.

Subset selection problems

For a given finite set I of size n, let 2^I denote the power set of I. A subset selection problem is an optimization problem of the following form:

$$\begin{aligned} \underset{\mathbf{x}\in 2^I}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p\,, \\ & \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\,. \end{aligned}$$

Equivalently, we can write:

$$\begin{aligned} \underset{\mathbf{x}\in \{0,1\}^n}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p\,, \\ & \mathbf{h}(\mathbf{x}) = \mathbf{0}_q\,. \end{aligned}$$

Here, the problem has been rewritten as an integer programming problem (as the domain is a subset of $\mathbb{Z}^n$). As a typical example, consider the knapsack problem. This problem is formulated as follows:

"*You have been given a set I of n items. Each item is represented by a vector*

(v_i, w_i), where v_i represents the monetary value of the ith item and w_i represents its weight. Additionally, you have been given a knapsack that has a weight limit of W. From all the items, select a subset that has the highest possible monetary value and respects the weight limit."

This problem can be translated into the following optimization problem.

$$\begin{aligned} \underset{\mathbf{x}\in\{0,1\}^n}{\text{minimize}} \quad & \sum_{i=1}^{n} x_i\, v_i \\ \text{subject to} \quad & \sum_{i=1}^{n} x_i\, w_i \leq W\,. \end{aligned}$$

Using this representation, a point $\mathbf{x}$ for which $x_i = 1$ represents a subset that includes the ith item, whereas $x_j = 0$ denotes that a subset does not include the jth item.

Solving subset selection problems

Unfortunately, even though the problem formulation above seems rather simple, finding an optimal point turns out to be hard for large n. This problem has an exponential time complexity. Because of that, people often resort to heuristics for solving this and related problems. More precisely, meta-heuristics such as ant colony optimization [39], genetic algorithms [40, 41] or tabu search [42] are frequently used to find (suboptimal) solutions of this type of problems. In Part II of this dissertation, we will develop a novel ant colony optimization procedure that can be used to solve subset selection problems. Moreover, we will show that this procedure has several interesting theoretical properties.

PART II

THE SELECTION OF AN OPTIMAL SUBSET OF MIXTURES

4 Scoring the subsets of a compositional dataset

4.1. Introduction

Modern high-throughput measuring equipment allows researchers to analyze large numbers of samples in a limited amount of time. The use of this equipment for the analysis of the (chemical) composition of mixtures often results in large databases containing compositional data. Even though the acquisition of these databases is generally interesting, it is often only a single step of an entire research project. In some cases, the post-processing of these samples (or the data obtained from them) is more expensive. For instance, a further analysis might call for a complex, time-consuming laboratory analysis, or the analysis of the resulting dataset needs to be performed by a numerical procedure that scales badly with the size of the dataset. Here, due to budgetary or time constraints, a researcher is sometimes forced to select a subset of the original set of samples (or data instances).

As a starting point for this chapter, we will assume that we have been given a collection of n mixtures. Each mixture is characterized by a d-part composition, representing the proportional contribution of d parts to the mixture. Subsequently, this characterization will be used to select a subset of size m (where $m < n$) from the collection of mixtures. More precisely, this characterization will be used to select a subset that optimizes a (diversity) criterion.

A small literature review shows that the problem setting described here (or at least strongly related problems settings) appears in several research disciplines. For example, in the field of bioinformatics the problem of 'core subset selection' exists of the selection of a subset of a genetic pool, *i.e.* a collection of (micro-) organisms, that maintains as much as possible of the genetic diversity present in the original collection. This problem has attracted a lot of attention during the last decades [43, 44, 45, 46, 47]. Naturally, the characterization of a micro-organism does not lead to a compositional dataset. In that sense the core subset selection is somewhat different from the problem that will be tackled in this chapter. However, due to its similarity, several ideas can be borrowed from this field. As a second example, in several branches of (bio-)chemistry, the problem of selecting a subset of a collection of samples that captures the variability of the entire collection appears frequently. Here, the goal often exists in reducing the collection of samples based on some high-throughput preliminary analysis, prior to more costly or time-consuming post-processing steps. As a result, subset selection procedures like the Kennard-Stone procedure [48] are implemented in popular chemometrics software packages such

as The Unscrambler® [49]. From a computational point of view, the selection of a representative subset of some collection can be interesting as well [50]. In applied machine learning, these procedures are sometimes used to split a given dataset in a train and a test set [51].

The examples above illustrate that the problem of selecting a subset of a collection of samples that optimally captures the diversity that is present in the original collection is rather common. Because of that, several procedures have been developed that allow to select such a subset. Especially in the field of applied biochemistry, there exist several applications that require the selection of a subset of *mixtures* from a large collection. In these cases, mixtures are sometimes represented by an enumeration of their components and a compositional vector indicating the proportional contribution of these components to these mixtures [52, 41]. Nevertheless, to the best of our knowledge, there exists no publication that explicitly takes the compositional nature of these data into account when selecting such a subset.

In this chapter, we address how the compositional nature of the data can affect such a selection. More precisely, by using some of the results introduced in the Chapter 2 of this dissertation, we will show that the procedures that respect the relative nature of the data can lead to subsets that strongly differ from the ones that are obtained by procedures that discard this property. Moreover, we will illustrate that most existing subset selection procedures can be tailored to handle compositional data in a sound manner.

The remainder of this chapter is organized as follows:

- In Section 4.2, score functions for the selection of subsets are introduced.
- In Section 4.3, the (optimization) problem of selecting an optimal subset is described.
- In Section 4.4, we experiment with several score functions.

4.2. Optimality criteria

4.2.1. Why do we need optimality criteria?

Naturally, when a collection of samples or the associated dataset needs to be reduced, we want this reduction to be optimal with respect to some criterion. Stated differently, the selected subset should be a 'good' subset with respect to this criterion. Several research papers in which (new) subset selection procedures are proposed or applied in a specific setting start by translating a 'good' subset as a subset that is:

> ... *diverse and representative* [53],
>
> ... *sampled with the goal of maximizing diversity and minimizing redundancy* [46],
>
> ... *a set of representative objects from a database* [52].

As can be seen from these examples, the general concept of a good subset is often described quite loosely. Therefore, in a second phase, these loose definitions are generally formalized. Mostly, the general idea of a subset that is representative or diverse is re-expressed as a subset that is *uniformly spread over the dataset*. Subsequently, some methods translate this requirement in a formal score function that unambiguously ranks all candidate subsets. Note that only a limited number of approaches actually define such score functions. For example the famous Kennard-Stone procedure [48] is only a heuristic that is claimed to lead to a representative subset of the data. In the original paper that introduces the Kennard-Stone procedure, there is no reference to a formal criterion that is optimized. In this chapter, we will mainly focus on procedures that do use a formal score function. Nevertheless, we will illustrate that other procedures can be modified to handle compositional data in a simple manner. Before formally introducing the type of score functions that we will be using, we end this paragraph by describing the general applicability of the methods we will be discussing.

Inevitably, the reduction of a data set will infer a loss of information. However, we can attempt to select a subset that minimizes the loss of information that is relevant w.r.t. a specific research goal. To be able to compare subsets, *i.e.* to be able to state that a given subset contains more relevant information than another subset, a score function can be constructed. Such a score function can be seen as an instrument that measures the amount of relevant information contained in a subset. It should be stressed that such a score function is highly application-dependent. Because of that, there exists probably no *general-purpose* score function that works well for all applications. Therefore, an objective function should always be chosen with the final research goal in mind. However, in this chapter we wíll consider general-purpose score functions. Therefore, the score functions described here should generally only be used if (1) a selection needs to be made prior to the description of research goals, or (2) the research goals are known but we lack the information to define a problem-oriented score function.

4.2.2. General-purpose score functions

Notational conventions

From this point on, we assume to have been given an (indexed) set $S = \{s_1, \ldots, s_n\}$ of n items. Moreover, it is assumed that each item is characterized by a (feature)

vector. For the ith item, this vector is denoted $\mathbf{z}_i \in Z \subseteq \mathbb{R}^d$ (Z is called the feature space of the items). The jth element of $\mathbf{z}_i$ is denoted $z_{i,j}$. These feature vectors can be collected in a matrix $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_n)^\top$. In summary, S is an indexed set of items (for example a set of mixtures) and $\mathbf{Z}$ is a matrix in which the ith row is the feature vector of the ith item in S (for instance a matrix of d-part compositions). Moreover, $\mathbf{Z}_{i,.}$ is the ith row of this matrix, $\mathbf{Z}_{.,j}$ is the jth column of this matrix and $\mathbf{Z}_{i,j}$ is the element in the jth column on the ith row.

A subset $\mathbf{x} \in 2^S$ can be written as an n-tuple $\langle s_1^{j_1}, \ldots, s_n^{j_n} \rangle$, where $j_i = 1$ if $s_i \in \mathbf{x}$ and $j_i = 0$ if $s_i \notin \mathbf{x}$. Let x_i be the i-th component of this n-tuple. We have that $x_i \in \{s_i^0, s_i^1\}$. Finally, the matrix $\mathbf{Z}^{\mathbf{x}}$ is a matrix that contains the feature vectors of the items in $\mathbf{x}$. Similarly, the matrix $\mathbf{Z}^{-\mathbf{x}}$ is a matrix that contains the feature vectors of the items that are not in $\mathbf{x}$.

It could be noted that this notation is somewhat heavy. However, we will be using (and extending) this notation in the following chapters.

General diversity measures

As good subsets are subsets that are diverse and representative for the collection, a score function can use measures of diversity. In some cases, existing diversity measures can be used. For example, for a core collection selection problem, the allelic richness of a candidate core can be used as a diversity measure [54]. Even though the allelic richness can be seen as a rather problem-specific score function, we feel that it fits within well within the class of general-purpose score functions. We motivate this as follows. Firstly, there is no concrete research question from which this objective (function) directly follows, and secondly, the authors explicitly express the hope that a selection based on this criterion will turn out to be useful for further activities such as breeding or long-term species survival. Another, perhaps slightly more widely applicable diversity measure is proposed in [41]. Here, the authors propose to use the following diversity criterion for a subset $\mathbf{x} \in 2^S$:

$$\min_j \left(\frac{\mathrm{var}(\mathbf{Z}^{\mathbf{x}}_{.,j})}{\mathrm{var}(\mathbf{Z}_{.,j})} \right) , \tag{4.1}$$

where $\mathrm{var}(\mathbf{Z}_{.,j})$ is the (sample) variance of the jth column of $\mathbf{Z}$. Formally, the function var is defined as:

$$\begin{array}{rccl} \mathrm{var}: & \mathbb{R}^n & \to & \mathbb{R} \\ & \mathbf{a} & \mapsto & \frac{1}{n}\sum_{i=1}^n a_i^2 - \left(\frac{1}{n}\sum_{i=1}^n a_i\right)^2 . \end{array}$$

Metric-based score functions

The *diversity* of a subset could be defined as an aggregation of the distances between the pairs of items within that subset. Similarly, the *representativity* of a subset w.r.t. the entire collection could be defined as the aggregated similarity between items that are included into the subset and items that are excluded from the subset. These aggregated similarities or dissimilarities can be used to define a score function. Therefore, we call these score functions metric-based score functions. It should be noted that this definition is new, *i.e.* we did not find publications that explicitly describe this strategy for constructing loss functions. However, some publications can be seen as examples of this general approach.

Formally, metric-based score functions use a metric (or a distance function) on pairs of feature vectors. Generally, a metric is a measure of the dissimilarity of two items. As the items are represented by their feature vectors, we need a metric on the feature space of the items. To construct a score function that is based on this metric, several strategies exist:

(i) The distance between each pair of items in the subset is computed and subsequently, these computed distances are aggregated using an aggregation function. For this aggregation, we can for instance use the mean, the minimum, the maximum , ... The Kennard-Stone procedure [48] can be interpreted as a heuristic that tries to optimize this measure (when the aggregation function is the minimum). Moreover, the approach used in [44] falls within this category.

(ii) For each item in S that is not in the subset, the distance to the nearest point in the subset is computed and subsequently, these computed distances can be aggregated. The k-means clustering algorithm (that is often used to select representative subsets [52]) can be seen as a heuristic that optimizes this criterion.

(iii) The principles described in (i) and (ii) are combined.

Using this characterization, strategy (i) could be used to maximize the diversity, and strategy (ii) could be used to maximize the representativity. However, the most important conclusion that can be drawn here is that metric-based score functions translate the problem of defining a score function into the problem of defining a metric. Fortunately, there exists a huge literature on metrics on a variety of feature spaces. As a simple example, when the feature space is $\mathbb{R}^d$, the Euclidean distance can easily be used. On the other hand, problem-specific knowledge may suggest other metrics.

Relationship between diversity criterion (4.1) and metric-based score functions

Interestingly, we could find a link between diversity criterion (4.1) and metric-based

score functions. This link is due to the following lemma.

Lemma 4.1. *For a given vector* $\mathbf{a} \in \mathbb{R}^n$*, we have that*

$$\mathrm{var}(\mathbf{a}) = \frac{1}{2\,n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (a_i - a_j)^2 \,.$$

Proof. Even though the proof is rather trivial, we present it here for completeness.

$$\begin{aligned}
\mathrm{var}(\mathbf{a}) &= \frac{1}{n} \sum_{i=1}^{n} a_i^2 - \left(\frac{1}{n} \sum_{i=1}^{n} a_i \right)^2 \\
&= \frac{1}{n} \left(\sum_{i=1}^{n} \frac{a_i^2}{2} \right) + \frac{1}{n} \left(\sum_{j=1}^{n} \frac{a_j^2}{2} \right) - \frac{1}{n^2} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} a_i \, a_j \right) \\
&= \frac{1}{n^2} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \frac{a_i^2}{2} + \frac{a_j^2}{2} \right) - \frac{1}{n^2} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} a_i \, a_j \right) \\
&= \frac{1}{n^2} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \frac{a_i^2}{2} + \frac{a_j^2}{2} - a_i \, a_j \right) \\
&= \frac{1}{2\,n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (a_i - a_j)^2 \,.
\end{aligned}$$

□

This lemma suggests that, when the feature space is $\mathbb{R}$, the variance of a set of observations is proportional to the sum of the squared distances between the observations. As a result, it is similar to the approach followed in strategy (i) of metric based score functions. Here, the squared difference can be used as a measure of dissimilarity between the points in the set and the aggregation functions can be the arithmetic mean. As a result, both score functions are identical (up to a scaling factor). Unfortunately, the squared difference is not a metric. Therefore, it is generally not recommended to be used as a measure of dissimilarity. This observation has two consequences: Firstly, it can help in gaining insight in diversity measures that use the variance. Secondly, as the squared difference (or more general squared Euclidean distances) is not a metric on scalars, we should be careful when looking at the variance as an aggregation of distances.

Diversity measures for compositional data

When a criterion is used to compute the diversity of a subset of compositional vectors, it should (according to the philosophy of compositional data analysis) respect the three main principles of compositional data analysis. From the discussion

in Chapter 2, it trivially follows that the diversity criterion given in Eq. (4.1) does not respect these principles. Therefore, it can be argued that this criterion is not appropriate here. However, we can derive a highly similar criterion that does respect these three principles. To obtain this measure, recall from Chapter 2 that, for a given compositional random vector $\mathcal{Y} = (\mathcal{Y}_1, \ldots, \mathcal{Y}_d)$, we defined the following matrix:

$$\mathbf{\Gamma}_{\mathcal{Y}} = \begin{pmatrix} \mathrm{var}(\ln(\mathcal{Y}_1/\mathcal{Y}_1)) & \cdots & \mathrm{var}(\ln(\mathcal{Y}_1/\mathcal{Y}_d)) \\ \vdots & \ddots & \vdots \\ \mathrm{var}(\ln(\mathcal{Y}_d/\mathcal{Y}_1)) & \cdots & \mathrm{var}(\ln(\mathcal{Y}_d/\mathcal{Y}_d)) \end{pmatrix}.$$

Moreover, the sum of all entries in this matrix can be used as a measure for the variability in Y. The manner in which the variability is characterized by this measure is highly similar to the manner in which the variability of a multivariate random vector (with sample space $\mathbb{R}^d$) is characterized. In the latter case, the trace of the covariance matrix is often used to characterize the variability. It can be proven [1] that the sum described above behaves in a similar manner. Naturally, we need a sample-based version of this measure. Using the notation introduced before, let $\mathbf{Y}$ be a $n \times d$ matrix, where the rows are d-part compositions, we define the matrix $\hat{\mathbf{\Gamma}}_Y$ as a $d \times d$ matrix

$$(\hat{\mathbf{\Gamma}}_{\mathbf{Y}})_{i,j} = \frac{1}{n}\sum_{\ell=1}^{n}(\ln(\mathbf{Y}_{\ell,i}/\mathbf{Y}_{\ell,j}))^2 - \frac{1}{n}\left(\sum_{\ell=1}^{n}(\ln(\mathbf{Y}_{\ell,i}/\mathbf{Y}_{\ell,j}))\right)^2.$$

It can easily be seen that $\hat{\mathbf{\Gamma}}_Y$ is an asymptotically unbiased estimator of $\mathbf{\Gamma}_Y$. The reasoning suggests that the diversity of a subset $\mathbf{x}$ can be defined as:

$$\sum_{i,j} \frac{(\hat{\mathbf{\Gamma}}_{\mathbf{Z}^{\mathbf{x}}})_{i,j}}{(\hat{\mathbf{\Gamma}}_{\mathbf{Z}})_{i,j}}.$$

Or, we could obtain a diversity criterion that is more similar to (4.1), as follows:

$$\min_{i \neq j}\left(\frac{(\hat{\mathbf{\Gamma}}_{\mathbf{Z}^{\mathbf{x}}})_{i,j}}{(\hat{\mathbf{\Gamma}}_{\mathbf{Z}})_{i,j}}\right).$$

Metric-based score functions for compositional data

From the discussion in Section 4.2.2, it follows that metric-based score functions for compositional data can easily be constructed by using a proper metric on compositional vectors and an aggregation function. From Chapter 2, we know that the Aitchison distance is a metric that respects the main principles of compositional data analysis. Even though there exists a vast number of aggregation functions that

can be used, we will be focusing on the minimum, the maximum and the arithmetic mean. Formally, we can construct score functions using the three strategies in Section 4.2.2.

(i) According to strategy (i), the score function assigns the following score to $\mathbf{Z}^{\mathbf{x}}$:

$$\min_{i \neq j} d_a(\mathbf{Z}^{\mathbf{x}}{}_{i,.}, \mathbf{Z}^{\mathbf{x}}{}_{j,.}) \,. \tag{4.2}$$

Note that the minimum can be replaced by another aggregation function.

(ii) According to strategy (ii), the score function assigns the following score to $\mathbf{Z}^{\mathbf{x}}$:

$$-\max_{j} \min_{i} d_a(\mathbf{Z}^{\mathbf{x}}_{i,.}, \mathbf{Z}^{-\mathbf{x}}_{j,.}) \,. \tag{4.3}$$

When using this function, the score of a subset $\mathbf{x}$ is determined by the point in $\mathbf{Z}^{-\mathbf{x}}$ that has the largest distance to any point in $\mathbf{Z}^{\mathbf{x}}$. Naturally, this distance should be as small as possible. However, to respect the convention that good subsets should have high score function values, a minus-sign was added. Here as well, other aggregation functions can be used.

4.3. Selecting an optimal subset

Once a score function has been chosen, the selection of an optimal subset can easily be translated into a mathematical optimization problem. For example, the search for the optimal subset of a fixed size k leads, using score function (4.2), to the following mathematical optimization problem.

$$\begin{aligned} \underset{\mathbf{x} \in 2^S}{\text{maximize}} \quad & \min_{i \neq j} d_a(\mathbf{Z}^{\mathbf{x}}_{i,.}, \mathbf{Z}^{\mathbf{x}}_{j,.}) \\ \text{subject to} \quad & |\mathbf{x}| = k \,. \end{aligned}$$

Moreover, this optimization problem is a specific example of the subset selection problem. Consequently, existing strategies for solving subset selection problems can be used to solve this problem. One of these strategies will be the topic of the following chapter. In the following sections, we elaborate on several of the issues that arise when trying to solve this problem.

4.3.1. Directly optimizing the score function

As the subset selection problem is a discrete optimization problem, an exhaustive search through the search space can be considered as a first approach to solve the problem. Unfortunately, when $|S|$ is large, an exhaustive search through 2^S cannot be performed within a reasonable amount of time. For example, in the last chapter

of this part of the dissertation, we consider a problem that requires a subset of 100 items to be selected from a set that contains 1000 items. An exhaustive search would require $6.3851 \cdot 10^{139}$ evaluations of the score function. Nevertheless, an exhaustive search can be interesting for small sets. We will use this approach to illustrate some properties of the score functions that were proposed earlier in this chapter.

As an exhaustive search through the search space is infeasible in most practical cases, we could attempt to construct a more efficient optimization algorithm that is capable of solving the problem. Unfortunately, it is unlikely that there exists an algorithm that can be used to solve the class of problems that we described within a reasonable amount of time[1]. As an alternative we could focus on constructing a good heuristic for this problem. The construction of such a heuristic will be the topic of the following chapter.

k-means procedure

We end this section by mentioning that the k-means procedure is an iterative procedure that attempts to maximize the following score function [55]:

$$f(\mathbf{x}) = -\sum_j \min_i (\|\mathbf{Z}^{\mathbf{x}}_{i,\cdot} - \mathbf{Z}^{-\mathbf{x}}_{j,\cdot}\|_2^2)\,.$$

However, there are no guarantees that this procedure will find an optimal point. Therefore, we generally refer to this procedure as a heuristic.

4.3.2. Surrogate problems and numerical recipes

Even though the approach that has been described before is a natural approach to solve sample selection problems, a considerable number of publications advocates the use of (what we loosely call) surrogate problems [52, 48, 50, 53]. In these publications, the sample selection problem is not solved directly. Instead, a different problem is defined and solved. Subsequently, it is hoped that the solution of this surrogate problem is a good solution to the original problem. Moreover, most publications do not explicitly state the objective function of the surrogate problem. Often, a numerical recipe is presented that is claimed to select a representative subset. Unfortunately, mostly there are close to no guarantees that the solution that is presented by these procedures will be close to the optimal subset according to a given score function. Nevertheless, some of these procedures, like the Kennard and Stone procedure [48] or the Optisim procedure [53] are used extensively and are generally assumed to be good subset selection procedures.

[1] However, for some specific cases, such an algorithm may exist.

Kennard and Stone procedure

The Kennard and Stone procedure aims at constructing a subset that covers the complete range of the original dataset in a uniform manner. However, the authors do not optimize a specific score function. Instead, they present a procedure that iteratively extends a given subset. As a starting point, the point that is closest to the multivariate arithmetic mean of the dataset according to some distance measure is selected. Note that this procedure thus requires a vector space structure on the sampling space. However, a simple modification of this starting point selection rule can relax that requirement. Subsequently, the procedure iterates as follows

1. From each point that is not in the subset, compute the distance to the subset (*i.e.* the distance to the closest point in the subset).

2. Add the point for which this distance is maximal and return to step 1.

Interestingly, this often celebrated procedure can be seen as a heuristic for the score function defined by (4.2). Therefore, the characteristics of solutions for that score function may be illustrative for solutions of the Kennard and Stone procedure as well. Moreover, using the vector space structure of the simplex described in Chapter 2, the Kennard and Stone procedure can be applied to compositional data. Lastly, when the Kennard-Stone procedure is used to select a subset of size k from a set of size n, the complexity is of order $\mathcal{O}(k\,n^2)$.

4.4. Experiments with different score functions

4.4.1. The Euclidean sample space

In this section, we illustrate that the choice of a particular metric-based loss function strongly influences the characteristics of the optimal subset. It was mentioned in Section 4.2.2 that we can generally distinguish between score functions that favor diversity (strategy (i)) and score functions that favor representativity (strategy (ii)). We illustrate the influence of this choice on an artificial dataset. We choose not to elaborate too strongly on this issue as there are several papers that indirectly compare these strategies as well. In these publications, Kennard and Stone and related selection procedures are compared with k-means like procedures. As the former can be seen as a heuristic for (strategy (i)) and the latter as a heuristic for (strategy (ii)), the conclusions we draw here are expected to be similar to the conclusions drawn in those papers [44, 52].

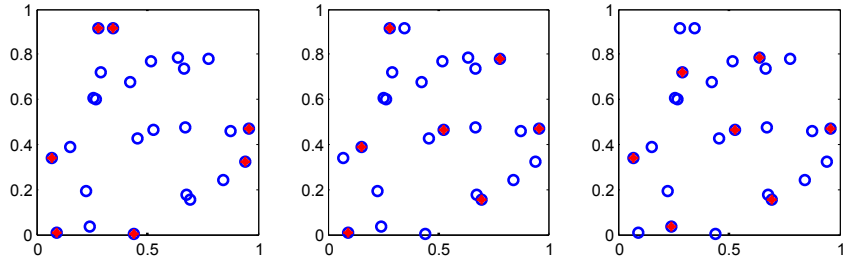


Figure 4.1: Scatter plot of the artificial dataset described in Section 4.4.1. The left panel shows the optimal subset using f_1 (selected points are indicated in red), the middle panel shows the optimal subset using f_2 and the right panel shows the optimal subset using f_3.

Optimal subsets were selected according to three score functions.

$$\begin{aligned} f_1(\mathbf{x}) &= \sum_{i,j} \left\| \mathbf{Z}^{\mathbf{x}}_{i,\cdot} - \mathbf{Z}^{\mathbf{x}}_{j,\cdot} \right\|_2 , \\ f_2(\mathbf{x}) &= \min_{i \neq j} \left\| \mathbf{Z}^{\mathbf{x}}_{i,\cdot} - \mathbf{Z}^{\mathbf{x}}_{j,\cdot} \right\|_2 , \\ f_3(\mathbf{x}) &= -\max_j \min_i \left\| \mathbf{Z}^{\mathbf{x}}_{i,\cdot} - \mathbf{Z}^{-\mathbf{x}}_{j,\cdot} \right\|_2 . \end{aligned}$$

Score functions f_1 and f_2 are examples of strategy (i), using different aggregation functions. Score functions f_3 is an example of strategy (ii).

An artificial dataset was generated by sampling 25 observations from a two-dimensional random vector that is uniformly distributed over $[0,1]^2$. Figure 4.1 shows scatter plots of the data and the optimal subsets. This example illustrates that maximizing the mean distance (f_1) between the points in the subset generally leads to a subset that is located near the convex hull of the dataset. Moreover, by using the minimum as aggregation function (f_2) instead of the sum, it seems the selected points are further apart. This is conform a remark in [44], where the authors conclude that a maximization of the minimum distance generally leads to a high average distance as well. However, the opposite is not generally true. Moreover, maximization of the minimum distance allows for the selection of points that are in the interior of the convex hull of the dataset. On the other hand, maximization of the sum of the distances between the observations in the subset encourages the selection of extreme points. This should not come as a surprise due to the link with the maximization of the sample variance given in Lemma 4.1. Lastly, as can be expected, strategy (ii) (using f_3) leads to a subset that is less focused on the extremes.

4.4.2. The (reduced) simplex sample space

In Chapter 2, it was argued that statistical methods that are used to analyze compositional data should respect the three main principles of compositional data. The methodology described in the current chapter suggests that interesting score functions can be obtained by considering metric-based score functions that uses the Aitchison distance. The resulting score functions are both intuitive and they respect the three main principles of compositional data analysis. In this section, we set up several (small-sized) sample selection problems to study the behaviour of these score functions. More precisely, we will compare the obtained subsets with those obtained using Euclidean distance.

An artificial dataset

As a first test case, we consider an artificial compositional dataset (the sample space is $\mathbb{S}_0^3$) that contains 24 points[2]. For this simulation experiment, we used the score function defined by (4.3). Moreover, when a problem has multiple optimal points, the optimal point with the smallest value for objective function (4.2) is visualized. Figure 4.2 shows the obtained subsets for subset sizes $k = 2, 4, 6, 8$ using the Euclidean distance and the Aitchison distance. From these plots, it can clearly be seen that the choice of the distance measure strongly influences the selection of a subset.

To explain this behaviour, we use Figure 4.3 that visualizes the artificial dataset. From a first visual inspection of these ternary plots, it can be suspected that these data are organized in two clusters. To see the influence of the distance measure used, consider Table 4.1 that reports the Euclidean and Aitchison distance between the points a, b, and c, d in Figure 4.3. For these points, it is clear that the Euclidean distance strongly differs from the Aitchison distance. For the Euclidean distance, a and b are farther apart than c and d. According to the Aitchison distance, c and d are farther apart. Knowing that the Aitchison distance only takes relative information into account, this can easily be explained. Both pairs mainly differ in the relative amount of component A. In an absolute manner, the difference between the relative amounts of component A is larger for couple (a, b) than for couple (c, d). However, using the relative difference, we obtain that c and d differ by a factor of 4 whereas a and b only differ a factor of 2. Similarly, using the Euclidean distance, the variability within the lower cluster (containing a and b) will be much higher than the variability in the upper cluster (containing c and d). Therefore, more points will be selected from this cluster. Using the Aitchison distance, the variability of both clusters will be more or less equal. Therefore, points from both clusters are selected.

[2] This dataset was generated by sampling from three Dirichlet distributions, the complete sampling procedure is described in Appendix 4.A.

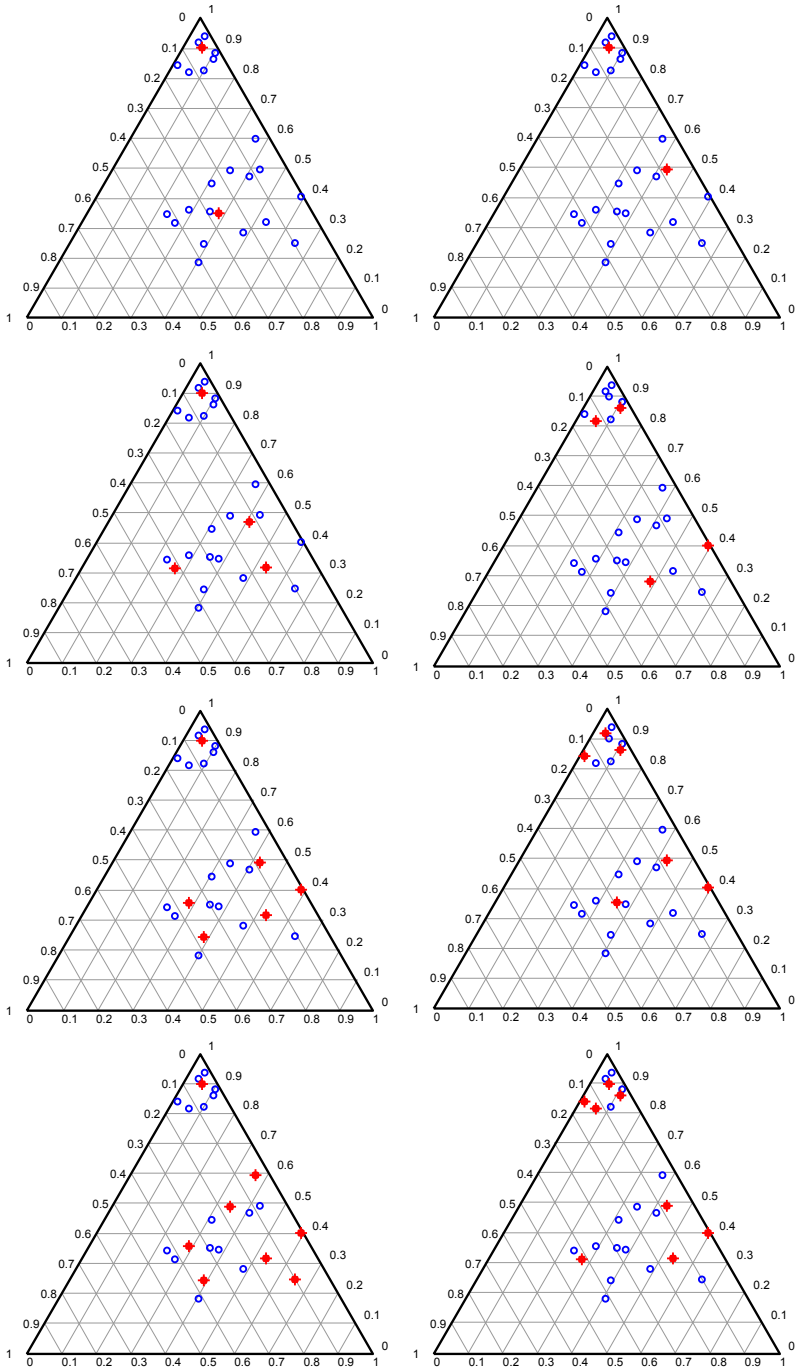


Figure 4.2: Ternary plot of the artificial dataset. Optimal subsets using score function $f(\mathbf{x}) = -\max_j \min_i \left\|\mathbf{Z}^{\mathbf{x}}_{i,.} - \mathbf{Z}^{-\mathbf{x}}_{j,.}\right\|_2$ (left column) versus $f(\mathbf{x}) = -\max_j \min_i d_a(\mathbf{Z}^{\mathbf{x}}_{i,.}, \mathbf{Z}^{-\mathbf{x}}_{j,.})$. The selected points are indicated in red. In the first (resp. 2nd, 3rd and 4th) row the subset size is $k = 2$ (resp. $k = 4, 6, 8$).

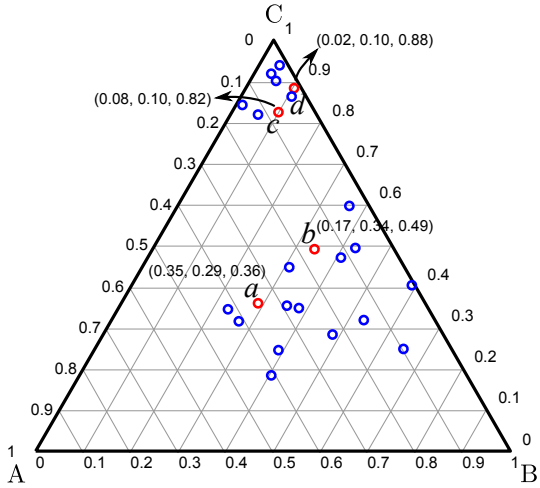


Figure 4.3: Ternary plot of the artificial dataset.

Table 4.1: Euclidean distance and Aitchison distance between the points a, b, and c, d in Figure 4.3.

couple	Euclidean distance	Aitchison distance
(a, b)	0.25	0.82
(c, d)	0.08	1.37

4.5. Concluding remarks

In this chapter, we illustrated that the often loosely described problem of sample subset selection can be formalized in several manners (Objective II.1). However, the manner in which this problem is translated into a mathematical optimization problem strongly influences the resulting subset. Firstly, when a metric-based score function is used, there is an influence of the strategy that is followed (diversity versus representativity). Secondly, when selecting a subset of compositional data, the Aitchison distance leads to subsets that strongly differ from those obtained using the Euclidean distance. However, it should be noted that the examples here mainly have an illustrative purpose. Therefore, the aim of this chapter is to illustrate, combine and link approaches from different areas of research.

The score functions that were introduced in this chapter can be expected to be hard to maximize efficiently. Therefore, in the following chapter, we resort to a meta-heuristic optimization strategy. However, provided that we can find a convex relaxation of one (or more) of these score functions, more efficient solution strategies (for example using B&B approaches) may be within reach. Unfortunately, it seems not trivial to come up with such a convex relaxation. The development of convex relaxations could be a suggestion for future research.

Finally, this chapter motivates the following chapter in which a meta-heuristic procedure will be developed that can be used to find high-quality solutions of the optimization problems that were encountered in this section.

4.A. Data generation procedure

To generate the data that is presented in Figure 4.3, 24 points were sampled from three Dirichlet distributions (8 observations from each distributions). The parameter vectors (s and $\boldsymbol{\alpha}$) are given hereafter:

- Distribution 1: $s = 35$, $\boldsymbol{\alpha} = (1/3, 1/3, 1/3)$,
- Distribution 2: $s = 35$, $\boldsymbol{\alpha} = (0.05, 0.45, 0.50)$,
- Distribution 3: $s = 35$, $\boldsymbol{\alpha} = (0.03, 0.03, 0.94)$.

5 An Ant Colony Optimization Algorithm for subset selection

5.1. Introduction

From the previous chapter, it is clear that the sample subset selection problem naturally leads to a mathematical optimization problem. Moreover, using the classification of Chapter 4, it can be seen as a specific case of the class of subset selection problems. Interestingly, it is well known that a lot of optimization problems in this class are very hard to solve (see for instance [56] for the example of feature-subset selection or [57, 58] for the knapsack problem, both problems are examples of subset selection problems that are hard to solve). Unfortunately, this also (probably) holds for (most of) the sample subset selection problems described in the previous chapter. As these subset selection problems are so hard to solve, there has been an interest in the past decades in developing heuristics that provide good but not necessarily optimal solutions to these problems. These heuristics range from problem-specific heuristics such as for instance the k-means procedure[1] to general meta-heuristics (such as genetic algorithms [59]) that are less problem-specific and can easily be adapted to solve a wide range of problems.

In this chapter, we study the applicability of the Ant Colony Optimization meta-heuristic (introduced in [60]), to solve subset selection problems. Moreover, as the sample subset selection problem is a special case of the subset selection problem, this study is relevant for solving sample subset problems as well. Interestingly, even though Ant Colony Optimization (ACO) has proven to be a useful heuristic for discrete optimization problems, such as the traveling salesman problem and several vehicle routing problems, it turns out that the basic variant of ACO performs very poorly on subset selection problems. This observation forms the main motivation of this chapter. By studying several properties of ACO for subset selection problems, we try to explain its inferior behavior. Moreover, these results will be used to present a modified ACO procedure that performs well in practice. The remainder of this section is organized as follows:

- In Section 5.2, the (traditional) application of ACO to subset selection is described.
- In Section 5.3, negative search bias in ACO is described.

[1] In the previous chapter, it was argued that the k-means procedure is a heuristic for the sample subset selection with a specific score function.

- In Section 5.4, bias-countering strategies are introduced and analyzed theoretically.
- In Section 5.5, we experiment with a novel procedure that implements bias-mitigating strategies in Ant System.
- In Section 5.6, the Max-Min Ant System is extended with bias-mitigating strategies.
- In Section 5.7, the special case of bias mitigation in sample subset selection problems is described.
- In Section 5.8, several conclusions are presented.

5.2. Ant colony optimization for subset selection

5.2.1. Notational conventions

As a starting point of this section, consider the general subset selection problem as defined in Chapter 3. For a given set $S = \{s_1, \ldots, s_n\}$ of n items and constraint functions $\mathbf{g} : 2^S \to \mathbb{R}^p$ and $\mathbf{h} : 2^S \to \mathbb{R}^q$, the following optimization problem is called a subset selection problem:

$$\begin{aligned} \underset{\mathbf{x} \in 2^S}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0}_p \,, \\ & \mathbf{h}(\mathbf{x}) = \mathbf{0}_q \,. \end{aligned}$$

However, in view of the goals of this chapter, this formulation is needlessly complicated. Instead, for a given triple (S, f, g) with item set S, objective function $f : 2^S \to \mathbb{R}$ and constraint function $g : 2^S \to \{0, 1\}$, the previous optimization problem can be rewritten as follows:

$$\begin{aligned} \underset{\mathbf{x} \in 2^S}{\text{maximize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & g(\mathbf{x}) = 1 \,. \end{aligned}$$

In this form, the problem consists of an objective function, and a single constraint function that determines whether a subset is feasible ($g(\mathbf{x}) = 1$) or infeasible ($g(\mathbf{x}) = 0$).

In the following sections, subsets of S will often be denoted as n-tuples, requiring an ordering/indexing of the elements of S, *i.e.* $S = \{s_1, \ldots, s_n\}$. Formally, a subset $\mathbf{x} \in 2^S$ can be written as an n-tuple $\langle s_1^{j_1}, \ldots, s_n^{j_n} \rangle$, where $j_i = 1$ if $s_i \in \mathbf{x}$ and $j_i = 0$ if $s_i \notin \mathbf{x}$. Let x_i be the i-th component of this n-tuple. We have

that $x_i \in X_i = \{s_i^0, s_i^1\}$. Denoting $X_{i]} = X_1 \times \ldots \times X_i$ $(i \leq n)$ we can write $2^S \equiv X_{n]} = X_1 \times \ldots \times X_n$. Using this tuple notation, we define the feasible space $X^{\mathrm{f}}_{n]} = \{\mathbf{x} \in X_{n]} \mid g(\mathbf{x}) = 1\}$. Now consider the set $S_i = S \setminus \{s_{i+1}, \ldots, s_n\}$ and $\mathbf{y}^i \in 2^{S_i}$. $\mathbf{y}^i$ can be written as an i-tuple: $\mathbf{y}^i = \langle s_1^{j_1}, \ldots, s_i^{j_i} \rangle \in X_{i]}$. Moreover, let $\mathbf{y}^i_{k]} = \langle s_1^{j_1}, \ldots, s_k^{j_k} \rangle$ be the first k $(k \leq i)$ components of the i-tuple $\mathbf{y}^i$, then $\mathbf{y}^i_{k]} \in X_{k]}$. For i-tuples, the feasible space is defined as $X^{\mathrm{f}}_{i]} = \{\mathbf{y}^i \in X_{i]} \mid (\exists \mathbf{x} \in X^{\mathrm{f}}_{n]})(\mathbf{x}_{i]} = \mathbf{y}^i)\}$. Finally, using the concatenation operator $\oplus$ on tuples, we can write $\mathbf{x} = \mathbf{x}_{i]} \oplus \langle s_{i+1}^{j_{i+1}}, \ldots, s_n^{j_n} \rangle$.

It should be recognized that this notation is rather heavy. Indeed, the notational conventions above extend the general notation used so far. However, to be able to describe our findings in a mathematically concise manner, the extension is necessary. Moreover, this notation can be seen as a compromise between the notation that is often encountered in the literature on ACO and the notation used in this dissertation.

5.2.2. Ant systems for subset selection

Ant System (AS) [60] was the first real ACO algorithm, inspired on the behavior of real ants. Since its development, this basic version of ACO has been altered numerous times in order to increase its performance or to adapt it to specific problem settings [61]. Although it is sometimes argued that the performance of AS is inferior to younger variants such as the Max-Min Ant System (MMAS) [62], we will mainly use it as the basic algorithm for this study. This choice seems justified since AS contains all the basic ACO elements. Interestingly, even in recent studies [63, 64], AS is still used and achieves state-of-the-art results. Additionally, it is the simplest form of an ACO algorithm. Moreover, rather than constructing a new state-of-the-art algorithm, our main goal is to study the behavior of ACO algorithms in general. Nevertheless, we can incorporate our methodology into ACO variants such as MMAS, as will be illustrated later.

Informal presentation of AS

The description of AS given here is (our interpretation of) AS for subset selection, closely following the original AS [60]. For the reader who is less familiar with ACO, we give an informal presentation of AS for subset selection in this section. This section briefly describes the general philosophy of AS, while the following section introduces AS in a formal manner. For this informal description, consider a knapsack problem[2] with item set $S = \{s_1, s_2, s_3\}$. The values and weights of these items are presented in Table 5.1. The capacity constraint equals 4.

[2] For a formal description of the knapsack problem see Chapter 3.

Table 5.1: Weights and values for a 0/1 unidimensional knapsack problem, with a capacity constraint that is equal to 4.

item	s_1	s_2	s_3
w_1, w_2, w_3	2	3	1
v_1, v_2, v_3	4	5	2

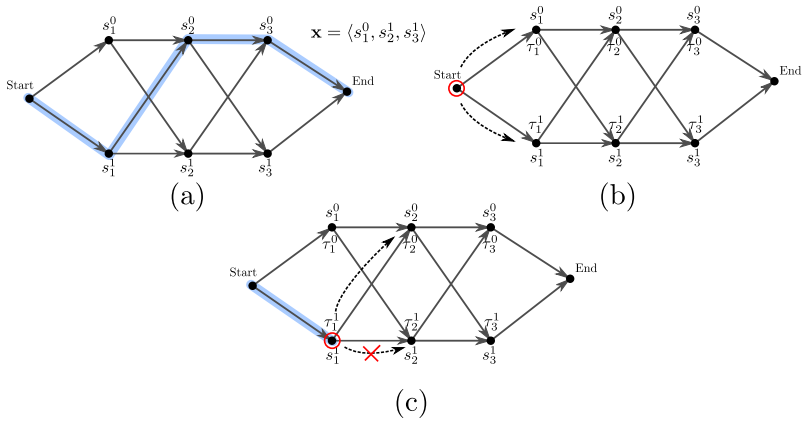


Figure 5.1: Panel (a) shows a graph representation of the subset selection problem. The path that is marked in blue represents the solution $\langle s_1^1, s_2^0, s_3^0 \rangle$. Panel (b) illustrates an ant that is located at the starting node of the graph. The partial solution of this ant is the empty subset $\langle\rangle$. This ant will use the values of the pheromone trail parameters τ_1^0 and τ_1^1 to decide whether or not to include the first item in the subset. This decision is based on a probabilistic decision rule that uses τ_1^0 and τ_1^1. Panel (c) illustrates an ant that has a partial solution $\langle s_1^1 \rangle$. Due to the capacity constraint, including item s_2 is not permitted. Therefore, the ant will be forced to extend its partial solution with component s_2^0 leading to $\langle s_1^1, s_2^0 \rangle$.

Intuitively, AS can be explained by representing this problem by means of a directed graph. Figure 5.1(a) shows a possible graph representation of this problem. Each path in this directed graph that starts at the 'start' node and ends at the 'end' node represents a (possibly infeasible) subset. The path that has been highlighted in blue represents the subset $\langle s_1^1, s_2^0, s_3^0 \rangle$. The objective value of this path is 2. Following the AS paradigm, multiple ants will construct solutions, *i.e.* paths on this graph. AS is an iterative procedure, therefore the ants are organized multiple generations. Typically, within each generation, individual ants will act independently. However, the paths that have been constructed by the ants in generation t will influence the paths that will be constructed by the ants in generation $t+1$. In the following paragraphs, we illustrate how individual ants construct a solution and how generations influence each other.

We start with the path construction procedure. An ant constructs a solution by incrementally constructing a path on the directed graph (therefore also called the construction graph). Such a path starts at the 'start' node that represents an empty partial solution $\langle\rangle$, see Figure 5.1(b) for an illustration. Subsequently, the ant makes a transition to one of the neighbouring nodes. For example, in Figure 5.1(c), a transition has been made from the start node to node s_1^1 representing that item s_1 has been added to the subset. This process is iterated until the 'end' node is reached.

During this construction procedure, an ant will use a series of probabilistic decision rules. These rules are based on so-called pheromone trail parameters. Thereto, each node is assigned a pheromone trail parameter. For example, for component s_2^1 this parameter is denoted τ_2^1. The value of this parameter during generation t is denoted $\tau_2^1(t)$. See Figure 5.1(b) for an illustration. The probabilistic decision rule states that an ant that is located at node s_i^j will select s_1^0 (resp. s_1^1) to extend its path with probability $\tau_1^0/(\tau_1^0+\tau_1^1)$ (resp. $\tau_1^1/(\tau_1^0+\tau_1^1)$). This principle is used iteratively until the 'end' node is reached.

To prevent infeasible solutions from being constructed, this probabilistic decision rule is slightly modified in some cases. For example, Figure 5.1(c) represents a partial solution $\langle s_1^1 \rangle$, the weight of this solution is 2. This means that, if item s_2 (that has a weight of 3) would be added, the weight of the partial solution would be 5. As this weight is in violation with the capacity constraint, this item is prevented from being selected.

During one generation, a (predetermined) number of ants will generate solutions using the construction procedure described before. These ants will act independently. Once all ants of generation t have generated a path, the pheromone trail parameters are updated. Firstly, all pheromone trail parameters are multiplied by a constant $(1-\rho)$, where $\rho \in [0,1]$ represents the evaporation rate. This operation mimics the evaporation of pheromones observed in nature. Secondly each ant will raise the values of some of the pheromone trail parameters. For example, an ant that

constructed the path $\mathbf{x} = \langle s_1^1, s_2^0, s_3^0 \rangle$ (the blue path) will update the parameters τ_1^1, τ_2^0 and τ_3^0. To each of these parameters, the value $f(\mathbf{x})$ will be added. Once this procedure is completed, the algorithm moves on to generation $t+1$ and the construction procedure recommences.

On average, the trail parameters that correspond to nodes that lie on paths (or solutions) with large objective values will (hopefully) receive an update that is higher than trail parameters of nodes that correspond to solutions with lower objective values. This means that high-quality solutions will have a higher probability of being generated than low-quality solutions. As a result, the search converges to regions in the search space that represent high-quality solutions. It should be noted that the wording used in this section reflects the origin of the AS as a procedure that is inspired on the behavior of real ants. Naturally, from an algorithmic point of view, this link is unnecessary.

Formal presentation of AS

We now present AS in a formal manner. As mentioned in the informal description, we assign a *pheromone trail parameter* τ_i^j to each s_i^j where $\tau_i^j(t) \in \mathbb{R}^+$ denotes the value of this parameter at generation (iteration) t. Moreover, we will use T to refer to the complete *matrix of pheromone trail parameters* (and $T(t)$ their values at time t). Since ACO is a constructive metaheuristic, each individual agent (ant) builds its own solution starting from scratch. As such, an agent starts with an empty partial solution (a tuple of length zero) $\mathbf{y}^0 = \langle\rangle$ and extends it through concatenation at each construction step. During the i-th ($i = 1, \ldots, n$) construction step, the tuple $\mathbf{y}^i \in X_{i]}$ is constructed as $\mathbf{y}^i = \mathbf{y}^{i-1} \oplus \langle s_i^j \rangle$ (with $j \in \{0, 1\}$ and $\mathbf{y}^{i-1} \in X^{\mathrm{f}}_{i-1]}$), where s_i^j is the result of a probabilistic decision rule, parameterized by T:

$$\Pr(s_i^j \mid \mathbf{y}^{i-1}, T(t)) = \frac{\tau_i^j(t)\eta(s_i^j)}{\sum\limits_{\{s_i^k \in E_{\mathbf{y}^{i-1}}\}} \tau_i^k(t)\eta(s_i^k)}, \quad \text{for all } s_i^j \in E_{\mathbf{y}^{i-1}}. \tag{5.1}$$

Here $\eta(s_i^j)$ represents the heuristic information, a simple (optional) weight that gives a rough estimate of the *a priori* desirability of adding this component given the current partial solution. $E_{\mathbf{y}^{i-1}} = \{\alpha \in X_i \mid (\exists \mathbf{x} \in X^{\mathrm{f}}_{n]})(\mathbf{x}_{i]} = \mathbf{y}^{i-1} \oplus \langle\alpha\rangle)\}$ contains the component values that can be used to extend the partial solution $\mathbf{y}^{i-1}$ such that $\mathbf{y}^i$ can still lead to feasible solutions. Consequently, we will denote the construction procedure as **SolutionConstruction**. Using Eq. (5.1), we have $\Pr(\mathbf{x} \mid T(t)) = \prod_{i=0}^{n-1} \Pr(x_{i+1} \mid \mathbf{x}_{i]}, T(t))$.

Once all ants within one generation have built their solution, each ant will individually update the pheromone trail parameters. In the t-th iteration, K ants construct

a set of n-tuples A_t. This set is used in the following update rule:

$$\tau_i^j(t+1) = (1-\rho)\,\tau_i^j(t) + \rho\,\frac{\sum\limits_{\{\mathbf{x}\in A_t \mid s_i^j \in \mathbf{x}\}} f^*(\mathbf{x})}{\sum\limits_{\{\mathbf{x}\in A_t\}} f^*(\mathbf{x})}\,, \tag{5.2}$$

where $f^*(\mathbf{x})$ is a monotone transformation of $f(\mathbf{x})$ (often $f^*(\mathbf{x}) = f(\mathbf{x})$). This update is performed for all pheromone trail parameters τ_i^j. As a consequence of this kind of update rule, the pheromones linked with solution components that were part of multiple solutions with high objective function values will receive high updates. Note that Eq. (5.2) is the update rule used in the hyper-cube framework (HCF) [65]. When the denominator in Eq. (5.2) is omitted, the standard AS update is obtained. This procedure will be denoted as **PheromoneUpdate**. Combining the principles described above leads to algorithm BasicAS.

```
procedure BasicAS(X_n], f, g)
    T(0) ←InitializePheromones
    while no convergence do                          ▷ terminated if converged
        solutionsLastGen ← NULL
        for ant = 1, ..., K do                       ▷ make paths
            x ← SolutionConstruction(T(t))
            solutionsLastGen.add(x)
        end for
        PheromoneUpdate(T(t), solutionsLastGen)
    end while
    return convergedSolution
end procedure
```

5.2.3. Why ACO?

As a meta-heuristic, ACO provides a flexible framework for solving discrete optimization problems. Once an ACO procedure is found that performs well for a particular problem, it is likely that only minor changes are needed to adapt that procedure to solve a new (yet related) problem [61, 66]. In Part II of this dissertation, we study such a setting. Different, yet related score functions need to be optimized. Clearly, other (less recent) meta-heuristics such as genetic algorithms or simulated annealing exhibit a flexibility that is similar to that of ACO. Given the successful application of ACO to a variety of discrete optimization problems [61] and the fact that ACO has not yet been applied extensively to subset selection problems, it seems an interesting algorithm to study. Nevertheless, in the following chapter, we will compare its performance with other heuristics.

5.3. Bias in metaheuristics

Search bias is a key concept in the field of evolutionary algorithms. Starting from an initial state, evolutionary algorithms (such as ACO or GAs) are expected to evolve towards the more promising regions in the search space. Typically, these algorithms start searching through the search space in a random, undirected manner. The solutions with the highest objective function values that are found during the first iteration will influence the search direction in the following iteration; they will bias the search towards the point in the search space they represent. This process continues as the solutions that are found during the second iteration will influence the search direction in the third iteration, and so on. As a result, the search might be drawn towards the promising regions in the search space. Stated differently, several positions in the search space will compete for attention from the algorithm. In order to find good solutions, the most attractive points in the search space should have the highest objective function values (and thus be good solutions). Unfortunately, the objective function values are not the only factors that influence the attractiveness of particular regions in the search space. Firstly, as we will see below, the search space can over-represent some solutions, making these solutions more attractive for the search procedure. Secondly, several properties of the evolutionary algorithm that is used can (unintentionally) direct the search towards specific regions in the search space. When such regions represent low-quality solutions, this type of bias can be harmful for an evolutionary algorithm. This phenomenon has been the subject of intensive study in GAs [67, 68, 69, 70] and is sometimes referred to as negative search bias. In the field of ACO, negative search bias has drawn some attention as well [65, 71, 72, 73]. In these studies, several (at least three) sources of bias have been identified (we review these sources below). Moreover, Merkle and Middendorf [74] define and use a deterministic model to study the dynamics of ACO. More precisely, they use a fixed point analysis of the deterministic model to explain several dynamical properties of ACO.

In the following sections, we give a short overview of several sources of bias that can occur in AS. However, as the problem representation is important w.r.t. the presence of bias, we briefly elaborate on that first.

5.3.1. Representations for subset selection

For a given subset selection problem, the presence of bias is heavily influenced by the definition of the search space and associated pheromone model. The version of AS described in the previous section has strong resemblance to the bit string representation (BSR) for subset selection problems introduced for GAs in [75], and further used by [76] in ACO. An n-tuple $\langle s_1^{j_1}, \ldots, s_n^{j_n} \rangle$ can be represented by the bit string $(j_1, \ldots, j_n)$. In terms of representation spaces, each solution is represented

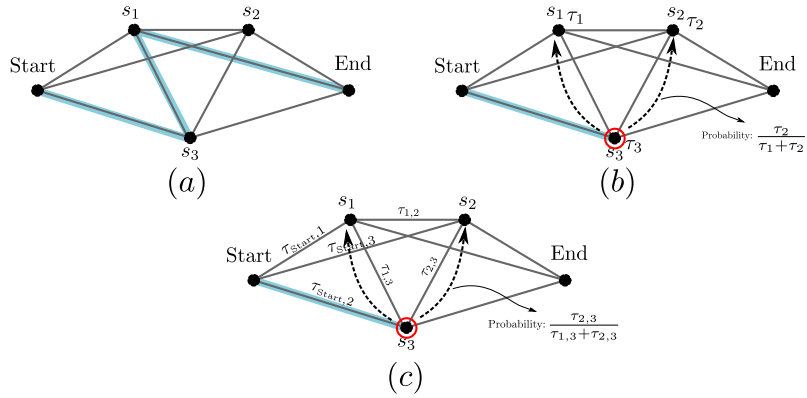


Figure 5.2: Panel (a) shows a graph representation of the knapsack problem in Table 5.1. This graph is undirected. The path highlighted in blue represents a subset that contains items s_1 and s_2. Panel (b) illustrates a pheromone model that associates pheromone trail parameters with the nodes of the graph. Here the pheromones represent the desirability of visiting a node. Panel (c) illustrates a pheromone model that associates pheromone trail parameters with the edges of the graph. Here the pheromones represent the desirability of selecting a specific edge.

by an n-tuple, which is equivalent to a bit string. As such, the search space is the set of all bit strings of length n. Note that any other ordering/indexing of the elements in S, *i.e.* $S = \{s'_1, \ldots, s'_n\}$, results in a corresponding $X'_{i\,]}$ and BSR. Other representations exist as well. In [40], the variable length representation (VLR) was introduced to solve the 0/1 unidimensional knapsack problem. The VLR has been used several times in the field of ACO [39, 71, 77]. In principle, during the construction procedure, an agent iteratively chooses an (unselected) item from S. This procedure leads to an ordered list of selected items. In this setting, typically one pheromone trail parameter is assigned to each item (this parameter is a measure for the desirability of selecting the item it is linked with). As such, the search space consists of a set of ordered lists of variable length. Figure 5.2 visualizes the construction graph of the VLR for the knapsack problem introduced earlier as well as two types of pheromone representations. For more details see the papers cited above.

Construction tree for the bit string representation

The graphical representations used before are generally interesting to gain some insight into the general strategy of ACO. However, the practical benefits of such representations are rather limited (at least w.r.t. this dissertation). Moreover, there exists another representation that will turn out to be more useful here. This representation is generally called the construction tree. In this binary tree, the root represents an empty partial solution. The leaves represent complete solutions. Each split represents a decision. One branch represents the selection of an item, the other branch represents its non-selection. Moreover, such a construction tree

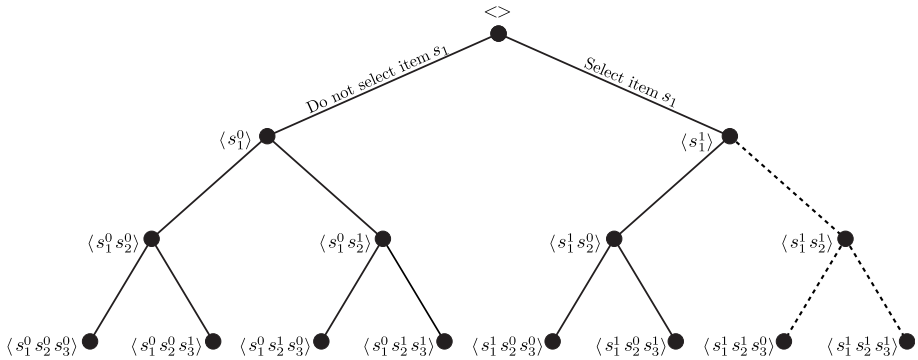


Figure 5.3: Construction tree of the knapsack problem in Table 5.1. The solution construction process can be visualized as a path that starts in the root node and progresses towards a leaf node in every iteration. Branches in the tree that lead to infeasible (partial) solutions are represented by dashed lines.

allows infeasible (partial) solutions to be represented graphically. Figure 5.3 shows a construction tree[3] of the knapsack problem from Table 5.1.

5.3.2. Sources of bias

Three main sources of bias have been described in literature. Firstly, in [71, 72, 73], the authors identify what they call *representation bias*. This type of bias emerges from the fact that (depending on the representation or pheromone model) a single solution $\mathbf{x} \subseteq S$ can be mapped upon several points in the representation space. As a consequence of the pheromone update procedure, solution components that are contained in solutions that are over-represented will be updated more frequently than other solutions. As a result, the search will be attracted by these over-represented solutions. ACO procedures that use the VLR are affected by representation bias. Indeed, as the construction procedure leads to an ordered list, each subset of S is represented by all possible permutations of its items. As the number of possible permutations of a subset increases with its cardinality, larger subsets will be over-represented compared to smaller subsets. On the other hand, when the BSR is used, each subset of S is represented by a single bit-string. As such, ACO procedures that use the BSR are not affected by representation bias.

Secondly, in Montgomery *et al.* [71], *construction bias* is identified as a consequence of problem constraints. Due to these constraints, it is stated that certain component choices lead to restrictions later in the solution construction process. This makes strongly constrained solutions more likely to be produced than unconstrained solutions. Both VLR and BSR are affected by this type of bias. From

[3] We say 'a' construction tree as each re-ordering of the items leads to a different construction tree.

the construction tree in Figure 5.3, it can be seen that solutions $\langle s_1^1, s_2^0, s_3^0 \rangle$ and $\langle s_1^1, s_2^0, s_3^1 \rangle$ are more heavily constrained than the remaining solutions. Indeed, when during a solution construction phase an ant has a partial solution $\langle s_1^1 \rangle$, it will be forced (in order to respect the capacity constraint) to proceed to node $\langle s_1^1, s_2^0 \rangle$ and its children $\langle s_1^1, s_2^0, s_3^0 \rangle$ and $\langle s_1^1, s_2^0, s_3^1 \rangle$. Therefore, these solutions are called constrained solutions.

In general, the probability of constructing a highly constrained solution is higher than the probability of constructing an unconstrained solution. This can easily be explained by means of the construction tree in Figure 5.3. Assume that during the solution construction procedure, at an unconstrained node, an agent decides to proceed to the left or right branch with equal probability. This means that solution $\langle s_1^0, s_2^0, s_3^0 \rangle$ will be constructed with a probability of 1/8. On the other hand, due to the presence of problem constraints, solution $\langle s_1^1, s_2^0, s_3^0 \rangle$ will be constructed with a probability of 1/4. As the constrained solution is constructed with a higher probability, its solution components will be updated with a higher probability. This will cause the search to be guided toward these constrained solutions.

Thirdly, Montgomery *et al.* [71] identify *assignment order bias.* The tuple notation, or equivalently the BSR, introduces an order on the items. Subsequently, this order is used during the solution construction phase. The order in which the items are considered can have an important influence on the probability that items are selected [71]. It should be noted, however, that this type of bias is strongly related to construction bias. To illustrate this, consider the 0/1 unidimensional knapsack problem. Here, the selection of items in the beginning of the construction procedure is unlikely to be heavily influenced by the capacity constraint. However, towards the end of the construction procedure, the problem constraints will prevent items from being selected. Therefore, both types of bias are heavily influenced by the problem constraints. As such, measures that mitigate the negative effect of the former are likely to (partly) resolve the latter too.

5.3.3. Detecting (harmful) bias

In the previous sections, different types and causes of bias were reviewed in the ACO framework. Although we focused mainly on subset selection problems, this negative bias can be present in other combinatorial optimization problems as well. Montgomery *et al.* [71] discuss whether bias can be expected to be present in some well-known combinatorial optimization problems based on problem-specific characteristics. An interesting tool to detect the presence of harmful bias is presented in [65, 73]. The authors use the evolution of the *expected iteration quality* as an indicator for the presence of a harmful bias. For an instance of a

combinatorial optimization problem and a given pheromone matrix $T(t)$ at iteration t, the expected iteration quality $W(T(t))$ is defined as:

$$W(T(t)) = \sum_{\{\mathbf{x} \in X^{\mathrm{f}}_{n\,]}\}} f(\mathbf{x}) \Pr(\mathbf{x} \mid T(t)) \,. \tag{5.3}$$

Here $\Pr(\mathbf{x} \mid T(t))$ is the probability of constructing n-tuple $\mathbf{x}$ given pheromone matrix $T(t)$. Moreover, the expected update is computed as

$$\tau_i^j(t+1) = (1-\rho)\tau_i^j(t) + \rho \frac{\sum\limits_{\{\mathbf{x} \in X^{\mathrm{f}}_{n\,]} \mid s_i^j \in \mathbf{x}\}} f^*(\mathbf{x}) \Pr(\mathbf{x} \mid T(t))}{\sum\limits_{\{\mathbf{x} \in X^{\mathrm{f}}_{n\,]}\}} f^*(\mathbf{x}) \Pr(\mathbf{x} \mid T(t))} \,, \tag{5.4}$$

where the denominator is only required in the HCF and typically $f^*(\mathbf{x}) = f(\mathbf{x})$. The authors argue that it is desirable that $W(T(t))$ is an increasing function of time. They link the absence of this characteristic to the presence of harmful bias.

Essentially, the formulas above describe the limit behavior of AS as the number of ants per iteration grows to infinity. Even though this limit case can be interesting to study the behavior of AS, there are no theoretical guarantees that this limit case is representative for a setting with a finite number of ants. This should be taken into consideration when judging an AS procedure by means of the expected iteration quality. As a result, we argue that the expected iteration quality of an AS procedure is an interesting characteristic. Moreover, an analysis of an AS procedure should incorporate, but not be limited to, an evaluation of the expected iteration quality.

5.4. Countering the bias

The presence of negative bias is acknowledged in many evolutionary algorithms. It has been indicated several times that the inclusion of bias countering strategies can (but does not necessarily) increase the performance of evolutionary algorithms [67, 78]. In the field of ACO, research on countering negative search bias is, as far as we know, limited to a few studies [71, 73, 77, 79]. An overview of some theoretical aspects about negative search bias can be found in [80]. These studies all adopt the same approach. Different representation spaces and pheromone models are proposed for the same problem, and based on mainly empirical results, the most appropriate among them is chosen. The experiments in these studies show that the choice of a specific representation can have an important influence on the performance of ACO. However, the authors mostly agree that none of the representations they propose is entirely free of bias. In the remainder of this section, we will formulate some basic ideas that can be considered to counter negative bias

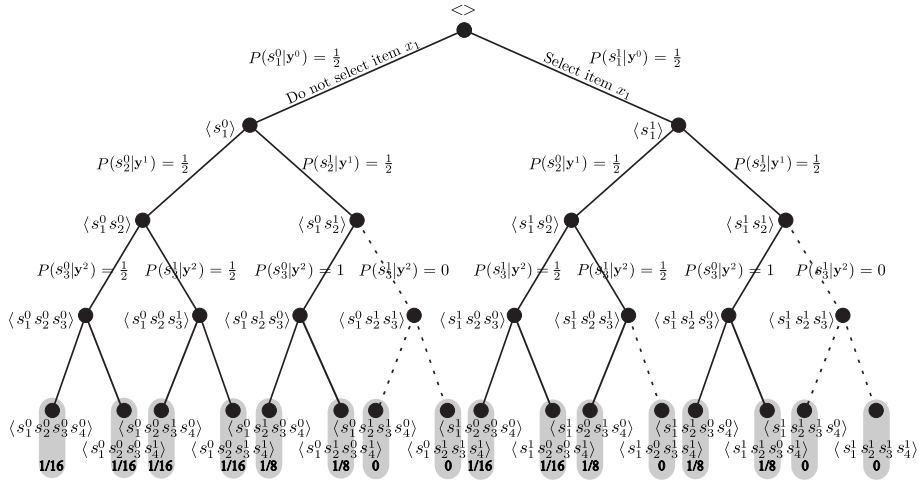


Figure 5.4: Construction tree of an artificial (toy) subset selection problem using the BSR. The n-tuples at terminal nodes represent complete solutions. The tuples at non-terminal nodes represent partial solutions. Solid edges represent paths that lead to feasible solutions, dashed edges represent infeasible paths. The conditional probabilities depicted near the edges are obtained with Eq. (5.1) using $\tau_i^j = c$ (with $c \in \mathbb{R}^+$ a constant representing the initial pheromone concentration) and discarding the heuristic information parameter. At each leaf, the probability that this leaf is reached using **SolutionConstruction** is given in bold. Note that, due to the presence of a construction bias, these probabilities are not equal.

in the AS algorithm applied to subset selection problems.

5.4.1. A modified update rule

As starting point, the BSR for a subset selection problem is chosen. Here, the search space can be visualized easily as a binary tree. As an introduction to our first bias-countering measure, consider the construction tree of an artificial (toy) subset selection problem depicted in Figure 5.4. An ant that has constructed the complete sequence $\mathbf{x} = \langle s_1^1, s_2^1, s_3^0, s_4^1\rangle$, will cause an update of the pheromones associated to all solution components that were selected. Even though component s_3^0 was merely added due to the problem constraints, it will still receive an update. Assuming equal objective function values for all feasible solutions, the expected update[4] for trail parameter τ_3^0 is the triple of the expected update received by s_3^1. Since s_3^0 and s_3^1 are competing for selection, s_3^0 will be favored in future iterations notwithstanding the fact that it does not lead to superior solutions. In this context

[4] The expected update is computed using Eqs. (5.1) and (5.4). No heuristic information is used to compute the expected pheromone update, which means that $\eta(s_i^j) = 1$ for all i and j. Moreover, all pheromone trail parameters are given identical values, *i.e.* $\tau_i^j(t) = c$ for all i and j.

we define *forced tuples.*

Definition 5.1. *For a given subset selection problem* (S, f, g) *and a search space* $X_{n]}$*, an* i*-tuple* $\mathbf{y}^i \in X^f_{i]}$ *is called a* ***forced tuple*** *(FT) if:*

$$\left| E_{\mathbf{y}^i_{i-1]}} \right| = 1\,,$$

where $\left| E_{\mathbf{y}^i_{i-1]}} \right|$ *denotes the cardinality of the set* $E_{\mathbf{y}^i_{i-1]}}$*.*

Using this definition, we denote $X^{\mathrm{FT}}_{i]} \subseteq X_{i]}$ as the set of forced i-tuples. When $\mathbf{y}^i$ is a forced tuple, one of the probabilities $\Pr(s^0_i \mid \mathbf{y}^i_{i-1]}, T(t))$ and $\Pr(s^1_i \mid \mathbf{y}^i_{i-1]}, T(t))$ (computed using Eq. (5.1)) equals zero, while the other equals one, due to the presence of problem constraints. In BasicAS, forced tuples can cause over-updates of some pheromone trail parameters. To prevent the effects of forced tuples from propagating through the algorithm, we propose a modification of the pheromone update rule. This modification simply consists of excluding components that were the result of forced tuples from the update procedure. Assuming that all solutions have the same objective function value, this will cause equal expected updates for components that compete for selection. This update procedure will be called **forcedUpdate** and leads to a new algorithm referred to as ForcedAS, which only differs from the original BasicAS in its update rule. More precisely, update procedure (5.2) is replaced with:

$$\tau^j_i(t+1) = (1-\rho)\,\tau^j_i(t) + \rho\,\frac{\displaystyle\sum_{\{\mathbf{x}\in A_t \mid s^j_i \in \mathbf{x} \wedge \mathbf{x}_{i]} \notin X^{\mathrm{FT}}_{i]}\}} f^*(\mathbf{x})}{\displaystyle\sum_{\{\mathbf{x}\in A_t \mid \mathbf{x}_{i]} \notin X^{\mathrm{FT}}_{i]}\}} f^*(\mathbf{x})}\,. \tag{5.5}$$

This procedure seems to be able to counter the propagation of the negative search bias (see experimental section). However, it does not prevent the agents from constructing some solutions more frequently than others. The probabilities given in Figure 5.4 remain valid. To counter this preferential behavior, a modified probabilistic decision rule is proposed in the next section. However, first we prove that the expected iteration quality $W(T(t))$ of ForcedAS is an increasing function of time. As such, it is illustrated that, in addition to its intuitive nature, ForcedAS has some appealing theoretical property. In this manner, we follow the approach of Dorigo *et al.* [65]. In [65], it was proven that for unconstrained problems the expected iteration quality of AS is an increasing function of time. Due to the presence of problem constraints, the proposition of Dorigo *et al.* does not apply to our setting. As we prove hereafter, ForcedAS provides a way to generalize the result of Dorigo *et al.* to problems with constraints. In this proof, we will be using the following lemma (see [81]).

Lemma 5.1. *Let $M \cup \partial M$ denote the manifold with boundary given by $x = (x_{i,j})$ where*

$$\left\{ x_{i,j} : x_{i,j} \geq 0 \text{ and } \sum_{j=1}^{q_i} x_{i,j} = 1 \right\}$$

where $q_1, \ldots, q_k$ is a set of nonnegative integers. Let W be a polynomial[5] *in the variables $\{x_{i,j}\}$, with nonnegative coefficients. Let h be a mapping $h : M \to M \cup \partial M$ defined by $y = h(x)$ where*

$$y_{i,j} = \frac{x_{i,j} \frac{\partial W}{\partial x_{i,j}}}{\sum_{k=1}^{q_i} x_{i,k} \frac{\partial W}{\partial x_{i,k}}} .$$

Then for any $x \in M$ and any $0 \leq \rho \leq 1$, it holds that

$$W(x) \leq W((1-\rho)\, x + \rho\, h(x)) .$$

Proposition 5.2. *Given a subset selection problem (S, f, g) and a search space $X_{n]}$, the expected iteration quality $W(T(t))$ of* ForcedAS *(with $f^*(\mathbf{x}) = f(\mathbf{x})$), is an increasing function of the iteration step t, more precisely*

$$W(T(t)) \leq W(T(t+1))$$

with equality if and only if $T(t) = T(t+1)$.

Proof. The expected iteration quality of ForcedAS for a given pheromone matrix $T(t)$ can be computed using Eq. (5.3), where $\Pr(\mathbf{x} \mid T(t))$ is computed using Eq. (5.1). Moreover, when we choose $T(t)$ such that $0 \leq \tau_i^0(t), \tau_i^1(t) \leq 1$ and $\tau_i^0(t) + \tau_i^1(t) = 1$, $(i = 1, \ldots, n)$, we have for all $\mathbf{x} = \langle s_1^{j_1}, \ldots, s_n^{j_n} \rangle \in X^{\mathrm{f}}_{n]}$

$$\Pr(\mathbf{x} \mid T(t)) = \prod_{i=1}^{n} \Pr(s_i^{j_i} \mid \mathbf{x}_{i-1]}, T(t)) \tag{5.6}$$

where for each $s_i^{j_i}$

$$\Pr(s_i^{j_i} \mid \mathbf{x}_{i-1]}, T(t)) = \begin{cases} \tau_i^{j_i}(t) & \text{, if } \mathbf{x}_{i-1]} \oplus \langle s_i^{j_i} \rangle \notin X^{\mathrm{FT}}_{i]}, \\ 1 & \text{, else.} \end{cases}$$

Since the expected pheromone update should exclude solution components that were the result of forced tuples, it is computed as

[5] The original theorem in [81] is only valid for homogeneous polynomials, however in the same publication the authors extend the theorem to nonhomogeneous polynomials.

$$\tau_i^0(t+1) = (1-\rho)\tau_i^0(t) + \rho \frac{\sum_{\{\mathbf{x}\in X^{\mathrm{f}}_{n\,]} | s_i^0 \in \mathbf{x} \wedge \mathbf{x}_{i\,]} \notin X^{\mathrm{FT}}_{i\,]}\}} f(\mathbf{x}) \Pr(\mathbf{x} \mid T(t))}{\sum_{\{\mathbf{x}\in X^{\mathrm{f}}_{n\,]} | s_i^0 \in \mathbf{x} \wedge \mathbf{x}_{i\,]} \notin X^{\mathrm{FT}}_{i\,]}\}} f(\mathbf{x}) \Pr(\mathbf{x} \mid T(t)) + \sum_{\{\mathbf{x}\in X^{\mathrm{f}}_{n\,]} | s_i^1 \in \mathbf{x} \wedge \mathbf{x}_{i\,]} \notin X^{\mathrm{FT}}_{i\,]}\}} f(\mathbf{x}) \Pr(\mathbf{x} \mid T(t))} . \quad (5.7)$$

The update formula for $\tau_i^1(t+1)$ is obtained in a similar way. Note that, when $0 \leq \tau_i^0(t), \tau_i^1(t) \leq 1$ and $\tau_i^0(t) + \tau_i^1(t) = 1$, the same will hold for their values at $t+1$. As such, they can be considered as probabilities (for an argumentation see [65]). Moreover, starting with a suitable $T(0)$, the expected evolution of $T(t)$ can be computed by iteratively using Eqs. (5.6) and (5.7).

Recall that $W(T)$ can be seen as a polynomial function of the variables $\tau_i^{j_i}$ (whereas $W(T(t))$ denotes the value of this function at time t). Using this notation, the partial derivative of $W(T)$ w.r.t. to τ_i^0 (and analogously for τ_i^1) at a given time t can be written as

$$\left.\frac{\partial W(T)}{\partial \tau_i^0}\right|_{T(t)} = \sum_{\{\mathbf{x}\in X^{\mathrm{f}}_{n\,]} | s_i^0 \in \mathbf{x} \wedge \mathbf{x}_{i\,]} \notin X^{\mathrm{FT}}_{i\,]}\}} f(\mathbf{x}) \frac{\Pr(\mathbf{x} \mid T(t))}{\tau_i^0(t)} .$$

When using $\rho = 1$, Eq. (5.7) can be reformulated as

$$\tau_i^0(t+1) = \frac{\tau_i^0 \left.\frac{\partial W(T)}{\partial \tau_i^0}\right|_{T(t)}}{\tau_i^0 \left.\frac{\partial W(T)}{\partial \tau_i^0}\right|_{T(t)} + \tau_i^1 \left.\frac{\partial W(T)}{\partial \tau_i^1}\right|_{T(t)}} .$$

Therefore, the update formula (Eq. (5.7)) can be identified with the function h in Lemma 5.1. This means that, from Lemma 5.1, we have that $W(T(t)) \leq W(T(t+1))$. The same proposition can be used to extend this result to other values of ρ. □

Proposition 5.2 can be generalized towards combinatorial optimization problems where $|X_i| > 2$. A typical example is the quadratic assignment problem [82]. As a result, our approach can be seen as a generalization of the proposition given in [65] to problems with constraints. This generalization is presented hereafter.

5.4.2. The expected iteration quality for assignment problems

Proposition 5.2 can be generalized to assignment problems (such as the quadratic assignment problem [83]) where $X_i = \{s_i^1, \ldots, s_i^{q_i}\}$ with $q_i \in \mathbb{N}$ (where the cardinality of this set was equal to two in Proposition 5.2). An n-tuple $\mathbf{x}$ with $x_i = s_i^j$ now represents a solution that assigns item s_i to group j (with $j = 1, \ldots, q_i$). Construction rule (5.1) can still be used, however, now $E_{\mathbf{y}^{i-1}} \subseteq X_i$. First, in analogy with Definition 5.1, we call the i-tuple $\mathbf{y}^i \in X^{\mathrm{f}}_{i]}$ a forced tuple if:

$$\left| E_{\mathbf{y}^i_{i-1]}} \right| < q_i \, .$$

Subsequently, we propose the following probabilistic decision rule to extend a partial solution $\mathbf{y}^{i-1}$ with a component $s_i^j \in E_{\mathbf{y}^{i-1}}$:

$$\Pr(s_i^j \mid \mathbf{y}^{i-1}, T(t)) = \begin{cases} \dfrac{\tau_i^j(t)}{\sum\limits_{\{s_i^j \in E_{\mathbf{y}^{i-1}}\}} \tau_i^j(t)} & , \text{ if } E_{\mathbf{y}^{i-1}} = X_i \, , \\ \dfrac{1}{\left|E_{\mathbf{y}^{i-1}}\right|} & , \text{ else.} \end{cases} \tag{5.8}$$

In analogy to the **forcedUpdate** procedure, a pheromone update procedure can be constructed as

$$\tau_i^j(t+1) = (1-\rho)\,\tau_i^j(t) + \rho \, \frac{\sum\limits_{\{\mathbf{x} \in A_t \mid s_i^j \in \mathbf{x} \wedge \mathbf{x}_{i]} \notin X^{\mathrm{FT}}_{i]}\}} f^*(\mathbf{x})}{\sum\limits_{\{\mathbf{x} \in A_t \mid \mathbf{x}_{i]} \notin X^{\mathrm{FT}}_{i]}\}} f^*(\mathbf{x})} \, . \tag{5.9}$$

The AS algorithm obtained by combining probabilistic decision rule (5.8) and update procedure (5.9) is a generalisation of ForcedAS to combinatorial optimization problems where $|X_i| > 2$. Moreover, for this generalized procedure as well, it can be proved that $W(T(t))$ is an increasing function of t. When applied to unconstrained problems (meaning $g(\mathbf{x}) = 1$ for all $\mathbf{x}$) this procedure reduces to the setting discussed by [65]. As such, it can be seen as a generalization of the setting in [65] to constrained optimization problems. Unfortunately, for some combinatorial optimization problems, this procedure can lead to inefficient algorithms. The main reason for this being that, when $|X_i|$ is large, it becomes likely that a lot of solution construction steps will lead to forced tuples and prevent updates from occurring. However, the ideas put forward here might serve as a basis for future research.

5.4.3. Modified probabilistic decision rule

In this section, a modified update rule is proposed that mitigates the preferential behavior towards highly constrained solutions. To present this modification more clearly, some new notations are introduced. For an i-tuple $\mathbf{y}^i$, denote $L_{\mathbf{y}^i} = \{\mathbf{x} \in X_{n\,]} \mid \mathbf{x}_{i\,]} = \mathbf{y}^i\}$ and $L^{\mathrm{f}}_{\mathbf{y}^i} = L_{\mathbf{y}^i} \cap X^{\mathrm{f}}_{n\,]}$. When representing the search space as a tree (e.g. Figure 5.4), $L_{\mathbf{y}^i}$ represents the set of terminal nodes (leaves) of the subtree rooted in $\mathbf{y}^i$. Using this notation, we have $L^{\mathrm{f}}_{\mathbf{y}^i} = L^{\mathrm{f}}_{\mathbf{y}^i \oplus s^0_{i+1}} \cup L^{\mathrm{f}}_{\mathbf{y}^i \oplus s^1_{i+1}}$. This notation is used in the following probability mass function, for any $\mathbf{y}^{i-1} \in X^{\mathrm{f}}_{i-1\,]}$:

$$\Pr\left(s_i^{j_i} \mid \mathbf{y}^{i-1}\right) = \begin{cases} \dfrac{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1} \oplus s^0_i}\right|}{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}}\right|} & \text{, if } j_i = 0\,, \\[2ex] \dfrac{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1} \oplus s^1_i}\right|}{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}}\right|} & \text{, if } j_i = 1\,. \end{cases} \tag{5.10}$$

Proposition 5.3. *Given a subset selection problem (S, f, g) and a search space $X_{n\,]}$, $\Pr(\mathbf{x}) = \prod_{i=i}^{n} \Pr(x_i \mid \mathbf{x}_{i-1\,]}) = \left|X^{\mathrm{f}}_{n\,]}\right|^{-1}$ for each $\mathbf{x} \in X^{\mathrm{f}}_{n\,]}$ using Eq. (5.10) to compute the conditional probabilities $\Pr(x_i \mid \mathbf{x}_{i-1\,]})$.*

Proof. First, we prove that the conditional probabilities computed using Eq. (5.10) can be used to define a (conditional) probability mass function on the solution components s^0_i and s^1_i. Consequently, we need to prove that for any $\mathbf{y}^{i-1} \in X^{\mathrm{f}}_{i-1\,]}$, we need that $\Pr(s^0_i \mid \mathbf{y}^{i-1})$ and $\Pr(s^1_i \mid \mathbf{y}^{i-1})$ are positive and that these probabilities add up to one.

Since Eq. (5.10) only uses ratios of cardinalities of sets, the result will be positive. Moreover, we have for each $\mathbf{y}^i \in X^{\mathrm{f}}_{i\,]}$

$$\frac{\left|L^{\mathrm{f}}_{\mathbf{y}^i \oplus s^0_{i+1}}\right|}{\left|L^{\mathrm{f}}_{\mathbf{y}^i}\right|} + \frac{\left|L^{\mathrm{f}}_{\mathbf{y}^i \oplus s^1_{i+1}}\right|}{\left|L^{\mathrm{f}}_{\mathbf{y}^i}\right|} = \frac{\left|L^{\mathrm{f}}_{\mathbf{y}^i}\right|}{\left|L^{\mathrm{f}}_{\mathbf{y}^i}\right|} = 1\,.$$

As a result, the multiplication rule can be used to compute $\Pr(\mathbf{x})$.

Next, we prove that $\Pr(\mathbf{x}) = \prod_{i=i}^{n} \Pr(x_i \mid \mathbf{x}_{i-1\,]}) = \left|X^{\mathrm{f}}_{n\,]}\right|^{-1}$. Substituting all conditional probabilities with Eq. (5.10) and $L^{\mathrm{f}}_{\mathbf{x}_{0\,]}} = X^{\mathrm{f}}_{n\,]}$ gives:

$$\Pr(\mathbf{x}) = \frac{\left|L^{\mathrm{f}}_{\mathbf{x}_{1\,]}}\right|}{\left|L^{\mathrm{f}}_{\mathbf{x}_{0\,]}}\right|} \times \frac{\left|L^{\mathrm{f}}_{\mathbf{x}_{2\,]}}\right|}{\left|L^{\mathrm{f}}_{\mathbf{x}_{1\,]}}\right|} \times \ldots \times \frac{\left|L^{\mathrm{f}}_{\mathbf{x}_{n-1\,]}}\right|}{\left|L^{\mathrm{f}}_{\mathbf{x}_{n\,]}}\right|} = \left|X^{\mathrm{f}}_{n\,]}\right|^{-1}\,,$$

which completes the proof. □

The probabilistic decision rule given in Eq. (5.10) can be incorporated in the AS solution construction procedure. Several ways exist to do so. However, the information obtained using Eq. (5.10) can be seen as some sort of domain knowledge or (variable) heuristic information. Because of this, the following combined probabilistic decision rule is proposed:

$$\Pr\left(s_i^{j_i} \mid \mathbf{y}^{i-1}, T(t)\right) = \begin{cases} \dfrac{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}\oplus s_i^0}\right| \tau_i^0(t)}{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}\oplus s_i^0}\right| \tau_i^0(t) + \left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}\oplus s_i^1}\right| \tau_i^1(t)} & , \text{ if } j_i = 0\,, \\[2ex] \dfrac{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}\oplus s_i^1}\right| \tau_i^1(t)}{\left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}\oplus s_i^0}\right| \tau_i^0(t) + \left|L^{\mathrm{f}}_{\mathbf{y}^{i-1}\oplus s_i^1}\right| \tau_i^1(t)} & , \text{ if } j_i = 1\,. \end{cases} \tag{5.11}$$

Eq. (5.1) is then replaced by Eq. (5.11) in the solution construction procedure. When $\tau_i^0 = \tau_i^1$, this procedure guarantees a completely uniform search over all feasible solutions. However, applying this modified construction procedure in combination with the standard pheromone update procedure can lead to an over-update of some pheromone trail parameters. As an example, consider the expected updates for τ_1^0 and τ_1^1 in the artificial (toy) problem (Figure 5.4). When $f(\mathbf{x}) = c$ for all $\mathbf{x} \in X^{\mathrm{f}}_{n]}$ and $\tau_i^0(0) = \tau_i^1(0)$, it is desirable that $\tau_1^0(1) = \tau_1^1(1)$ since there is no reason to prefer s_1^0 over s_1^1. Nevertheless, the expected values for $\tau_i^0(1)$ and $\tau_i^1(1)$ (resp. 6/11 and 5/11 when $\rho = 1$) computed using Eq. (5.4) in combination with Eq. (5.11), are not equal. In order to counter this over-update, guided tuples are defined as a generalization of forced tuples.

Definition 5.2. *For a given subset selection problem (S, f, g) and a search space $X_{n]}$, an i-tuple $\mathbf{y}^i \in X^{\mathrm{f}}_{i]}$ is called a* ***guided tuple*** *(GT) if:*

$$\left|L^{\mathrm{f}}_{\mathbf{y}^i_{i-1]}\oplus s_i^0}\right| \neq \left|L^{\mathrm{f}}_{\mathbf{y}^i_{i-1]}\oplus s_i^1}\right| .$$

For example, in Figure 5.4, the partial solution $\mathbf{y} = \langle s_1^1 \rangle$ is a guided tuple as $\left|L^{\mathrm{f}}_{s_1^0}\right| = 6$ and $\left|L^{\mathrm{f}}_{s_1^1}\right| = 5$. Essentially, guided tuples are the result of a propagation of the problem constraints towards the start of the solution construction procedure (*i.e.* the top of the construction tree). They are linked with probabilistic decisions that were influenced by the (Hamming) distance from the current partial solution to the boundary of the feasible search space. The probabilistic decision rule given by Eq. (5.11) uses this information to reduce the influence of the problem constraints. Notably, the modified probabilistic decision rule only differs from the original decision rule at guided tuples.

The modified probabilistic decision rule (Eq. (5.11)) ensures that the search space

is sampled in a uniform manner. However, as illustrated earlier, this decision rule may cause an imbalance in the pheromone update. To restore this imbalance, a modification of **forcedUpdate** is proposed. The principle of the update procedure remains, the only change concerns the size of update. The construction of a tuple $\mathbf{x}$ with $x_i = s_i^j$ will cause an update of τ_i^j of size

$$f^*(\mathbf{x}) = 2\left(1 - \frac{\left|L^{\mathrm{f}}_{\mathbf{x}_{i]}}\right|}{\left|L^{\mathrm{f}}_{\mathbf{x}_{i-1]}}\right|}\right) f(\mathbf{x})\,. \tag{5.12}$$

The update rule referred to as **guidedUpdate**, consists of the standard update rule given in Eq. (5.2) where $f^*(\mathbf{x})$ is computed using Eq. (5.12). Applying the modified solution construction procedure and update rule in AS leads to an algorithm that we will refer to as GuidedAS. Moreover, from this discussion, the following lemma follows directly.

Proposition 5.4. *Given a subset selection problem (S, f, g) and a search space $X_{n]}$, and $f(\mathbf{x}) = c$ for all $\mathbf{x} \in X^{\mathrm{f}}_{n]}$. For an initial pheromone matrix $T(0)$, where $\tau_i^0(0) = \tau_i^1(0)$ for $i = 1, \ldots, n$, we have $\tau_i^0(t) = \tau_i^1(t)$ for all t when τ_i^j is computed using Eq. (4) in combination with Eqs. (5.11) and (5.12).*

From this proposition, we have that, in the (artificial) situation where all solutions have equal objective function values, GuidedAS will cause equal expected pheromone updates for each pair of (competing) components s_i^0 and s_i^1. As such, GuidedAS exhibits no preferential behavior towards specific components.

5.4.4. Towards a working algorithm

As described above, GuidedAS requires knowledge of the construction tree to compute Eqs. (5.11) and (5.12). At worst, an enumeration of the entire search space will be needed to calculate the required probabilities. However, as will be shown in the application section, for some subset selection problems these probabilities can be obtained efficiently without such an enumeration. For other subset selection problems, we resort to an approximation of the construction tree to compute Eqs. (5.11) and (5.12). A simple heuristic that can be used to approximate the tree is illustrated in Figures 5.5 and 5.6. Here, it is initially assumed that $X^{\mathrm{f}}_{n]} = X_{n]}$. As an agent discovers infeasible regions, the presence of these regions will be taken into account when computing (5.11) and (5.12) for all future agents. The location of these infeasible regions can be stored efficiently in a data structure that is generally known as a 'binary tree'. This data structure strongly reduces the computational overhead. Moreover, during a single run of GuidedAS only a small part of the search space is actually explored. Consequently, only a small part of the (feasible) search space will have to be stored in memory. This keeps the memory requirements within acceptable limits. It should be noted that this

heuristic can be adapted towards specific subset selection problems. In case of the knapsack problem, the typical symmetric pattern within the construction tree can be used to get an improved estimate of the tree structure.

5.4.5. An analysis of bias-mitigating strategies

In the previous sections, a set of bias-mitigating measures has been proposed. These measures were constructed with the goal in mind that, when $\tau_i^j = c$, the (expected) update received by any component, or pheromone trail parameter associated to it, should only be affected by the average objective function value of all solutions this component is part of. More precisely, the ratio of the updates received by (two) components (s_i^1 and s_i^0) that compete for selection should only depend on the ratio of the average objective function value of the solutions that contain one of these solution components. GuidedAS enforces this property by successively forcing a uniform distribution of the feasible solution sequences through a modified construction rule, and applying an appropriate update rule. Even though these principles do not explicitly eliminate the sources of bias discussed in Section 5.3.2, they were designed to eliminate the propagation (possibly negative) of this bias throughout the pheromone model.

Since GuidedAS uses the BSR, a one-to-one mapping exists between n-tuples $\mathbf{x} \in X_{n]}$ and subsets $\mathbf{x} \in 2^S$. As such, representation bias does not exist. Construction bias on the other hand, will be present due to problem constraints. An important consequence of this type of bias is the systematic over-update some components receive. From Proposition 5.4 it follows that GuidedAS counters this consequence. As a final type of bias, we consider assignment order bias. Since the BSR requires a specific ordering to traverse the decision variables, this ordering might influence the probability that the algorithm will converge to a specific solution [71]. Here, we define assignment order bias as a property of an AS algorithm applied to a subset selection problem.

Definition 5.3. *The application of an AS algorithm to a subset selection problem (S, f, g) is affected by assignment order bias if there exist at least two bit string representations BSR1 and BSR2 (where these representations only differ in the ordering that is used) such that*

$$\left(\exists\, \mathbf{x} \subseteq S\right) \left(\hat{p}_1(\mathbf{x} \mid T(0)) \neq \hat{p}_2(\mathbf{x} \mid T(0))\right),$$

where $\hat{p}_1(\mathbf{x} \mid T(0))$, resp. $\hat{p}_2(\mathbf{x} \mid T(0))$, denotes the probability that AS will converge to solution $\mathbf{x}$ using BSR1, resp. BSR2, for a given initial pheromone matrix $T(0)$.

Simple simulation experiments show that the assignment order can have a strong influence on the performance of BasicAS when using the BSR. This finding is

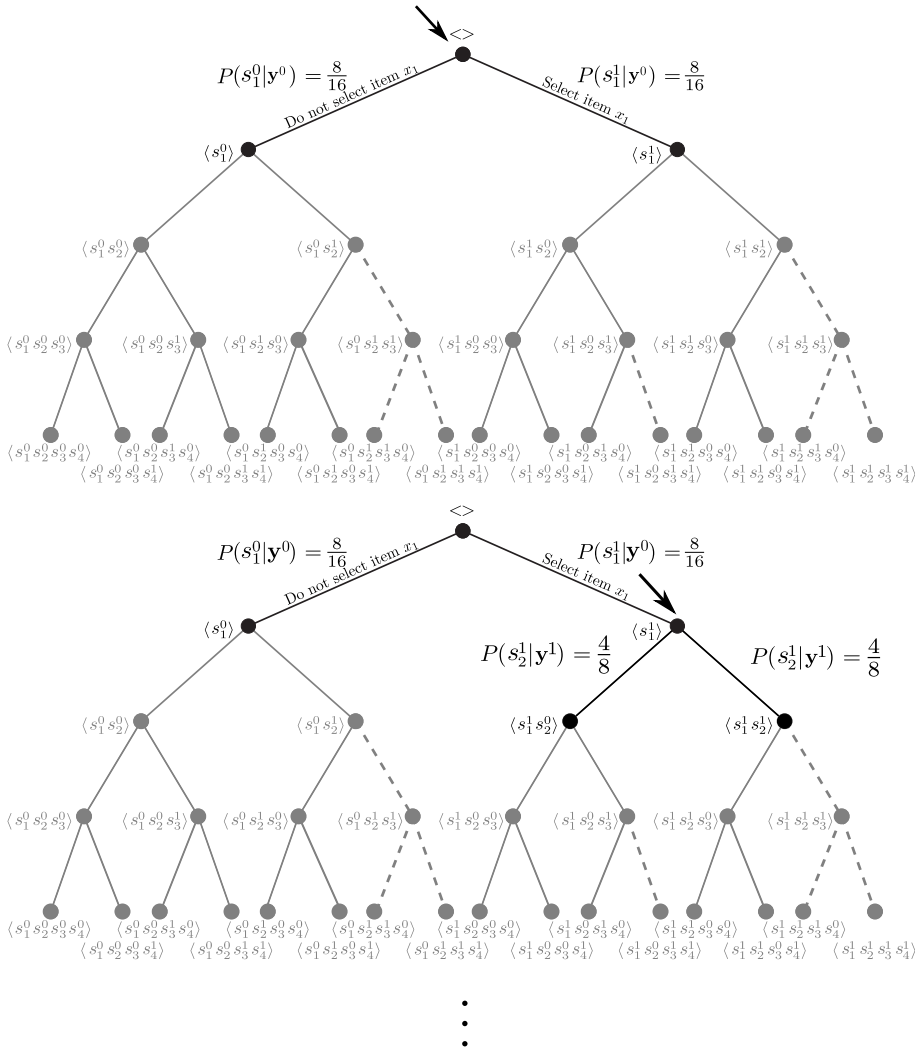


Figure 5.5: Use of a heuristic to approximate the construction tree (part one). The arrow indicates the partial solution in the memory of the ant. Initially, all solutions are assumed to be feasible. As the solution is extended, the tree's structure is partially discovered. By back propagation, the estimates of the number of feasible leaves under each of the nodes and associated probabilities are updated. The explored part (denoted in black) is actually stored in memory. The gray, unexplored part is not stored.

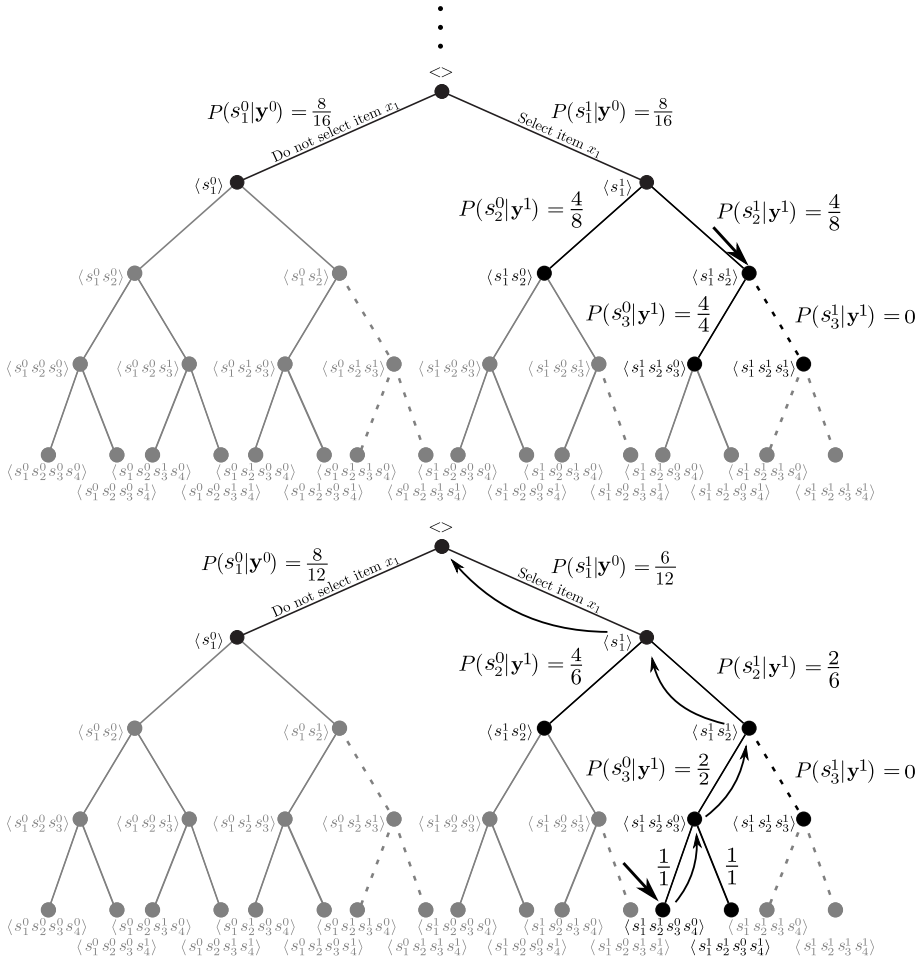


Figure 5.6: Use of a heuristic to approximate the construction tree (part two).

strengthened by several studies arguing that some orderings can have a positive effect on the performance of AS in case of quadratic assignment problems. It can be shown (empirically) that GuidedAS is affected by assignment order bias as well. However, as illustrated in the experimental section, its effects are strongly mitigated.

As pointed out in [71], assignment order bias and construction bias are closely related. In the presence of constraints, the ordering in which the items are traversed influences the intensity of the search in several parts of the search space. Different orderings will result in different construction trees favoring different parts of the search space. As such, the ordering can be used to bias the search towards promising regions in the search space. Moreover, it has been argued that randomly permuting the items each time an agent constructs a solution sequence acts as a bias-mitigating measure. We shall refer to this procedure as PermutedAS. It turns out that GuidedAS and PermutedAS have several similarities in the way they try to mitigate search bias. Firstly, from Proposition 5.3 it follows that, when $\tau_i^0 = \tau_i^1$, the probabilistic decision rule (5.11) will produce a uniform distribution (*i.e.* the distribution with maximal entropy) over all feasible solutions. For any fixed ordering of the items, the distribution implied by BasicAS and decision rule (5.1) will be non-uniform and have a lower entropy. When applying PermutedAS, the resulting distribution can be obtained by averaging over all orderings, often resulting in an increased entropy. Secondly, from Proposition 5.4 it follows that, when all objective function values are equal, there is a balanced (expected) update between solution components that compete for selection. Even though PermutedAS cannot cause these expected updates to become completely equal, its averaging effect has the tendency to level out these updates. Taking this similarity into account, it might be argued that PermutedAS can be used as an efficient alternative to GuidedAS. However, as we will illustrate in the application section, problems might occur when the probability of constructing a specific solution sequence is too heavily influenced by the ordering of the items. Since the behavior of individual agents that are confronted with the same pheromone matrix will be very diverse, convergence problems might occur. Moreover, it might take numerous agents per generation to obtain the averaging behavior that is needed to establish the properties described above.

5.5. Algorithmic performance: experimental setup

In this section, we illustrate how bias-avoidance mechanisms affect the performance of AS for subset selection problems. To be able to study the effects caused by negative bias, we choose not to extend the basic AS with frequently used elitism-like techniques here (in Section 6, we will embed our bias-mitigating strategies within the Max-Min Ant System [62] to improve performance). The 0/1 unidimensional

knapsack problem will often be used as a reference. Formally, this knapsack problem can be seen as a subset selection problem (S, f, g) where each item s_i can be represented by a couple $(w_i, v_i) \in \mathbb{R}^{+2}$, where $f(\mathbf{x}) = \sum_{\{s_i \in \mathbf{x}\}} v_i$, and $g(\mathbf{x}) = \left[\sum_{\{s_i \in \mathbf{x}\}} w_i \leq C\right]$, with $C \in \mathbb{R}^+$ the capacity constraint and [.] the indicator function.

5.5.1. Implementation of AS variants

To be able to experiment with the algorithmic ideas introduced earlier, a Java (Java SE 6) library was developed that includes implementations of BasicAS, ForcedAS, GuidedAS and GuidedAS with the construction tree heuristic as well as a number of facilities that allow the experiments in the following sections to be performed. This library allows (an approximation of) the construction tree of a subset selection problem to be generated in a dynamical manner. The building block of this implementation is a set of classes that implement a binary tree data structure.

5.5.2. Unpreferential behavior: an example

ForcedAS and GuidedAS were developed to mitigate the (undesired) preferential behavior of BasicAS towards some solutions. To illustrate the effect of the modifications that were proposed, consider the (artificial) subset selection problem depicted in Figure 5.4. In a first experiment, all (feasible) solution sequences were given objective function value 10. Each variant of the AS algorithm was run 1000 times (with an initial pheromone concentration of 100, $\rho = 0.3$, and we chose as many ants as there are items, *i.e.* 4 ants per generation) until convergence. Individual runs were considered to have converged once a particular solution sequence appeared at least 18 times in a series of 20 successively constructed solution sequences. Figure 5.7 (a, d, g, k) represents the probability of convergence to each of the feasible solution sequences (based on these 1000 runs). Since all objective function values are equal, it is desirable that there is no preferential behavior towards specific solutions. It can be seen in Figure 5.7 (a, d, g, k) that GuidedAS gives the best approximation to a uniform distribution over the feasible solution sequences, indicating that this algorithm is probably the least influenced by (negative) search bias. Moreover, BasicAS shows a strong preference towards some solution sequences, whereas ForcedAS shows intermediate behavior. PermutedAS leads to a smoother histogram than BasicAS, however, it shows a bias towards $\langle s_1^0, s_2^0, s_3^0, s_4^0 \rangle$. This can be explained by observing that, within the set of feasible 4-tuples, for each i, s_i^0 is more (or equally when $i = 3$) abundant than its competitor s_i^1, causing higher updates for τ_i^0. On the other hand, preferential behavior towards other components will be mitigated by the random permutations.

Table 5.2: Weights and values for two examples of the 0/1 unidimensional knapsack problem, both with a capacity constraint of 10.

item	s_1	s_2	s_3	s_4
$w_1, \ldots, w_4$	3	5	6	2
$v_1, \ldots, v_4$ for problem 1	4	5	9	1
$v_1, \ldots, v_4$ for problem 2	3	5	5	4

As a second experiment, consider a 0/1 unidimensional knapsack problem where S contains four items with weights, values and capacity constraints given in Table 5.2. Note that the construction tree of both problems is identical to the one in Figure 5.4. However, the objective function values of the solution sequences are different here. As can be seen from Figure 5.7 (b,e,h,j), all variants converge to high-quality solutions. It can be seen that the probability that BasicAS converges to the optimal solution is heavily influenced by the location (with respect to infeasible solutions) of this optimum in the construction tree. ForcedAS and GuidedAS seem to be less influenced by this location. PermutedAS is able to mitigate the effects of BasicAS for the first knapsack problem, however, its performance is inferior to the other modified AS versions here. Moreover, when the optimal region is located in a region that is favoured by BasicAS, it outperforms the other variants.

5.5.3. Experiments on knapsack problems

In [65], the authors argue that it is desirable that the expected iteration quality $W(T(t))$ of a combinatorial optimization problem is an increasing function of time. They relate a (temporary) drop in the expected iteration quality to the presence of negative search bias. For ForcedAS, we have proven that the expected iteration quality is an increasing function of time. However, we were unable to extend this idea to GuidedAS. To examine the evolution of the expected iteration quality here, 30 000 instances of the 0/1 unidimensional knapsack problem were created following the method of [84] with a size ranging from 5 to 20 items. For each of these instances, the expected iteration quality of GuidedAS increased as a function of t. Depending on the tightness ratio (which is the ratio of the capacity of the knapsack and the added weights of all items, this ratio ranged between 0.25 and 0.50) used to create these instances, the expected iteration quality of BasicAS exhibited temporary drops in 5 - 50 % of the cases.

The toy examples given before illustrate the negative influence that bias can have on the performance of AS. Experiments with knapsack problems (generated according to [84]) indicate that our bias-avoidance mechanisms have a positive influence on

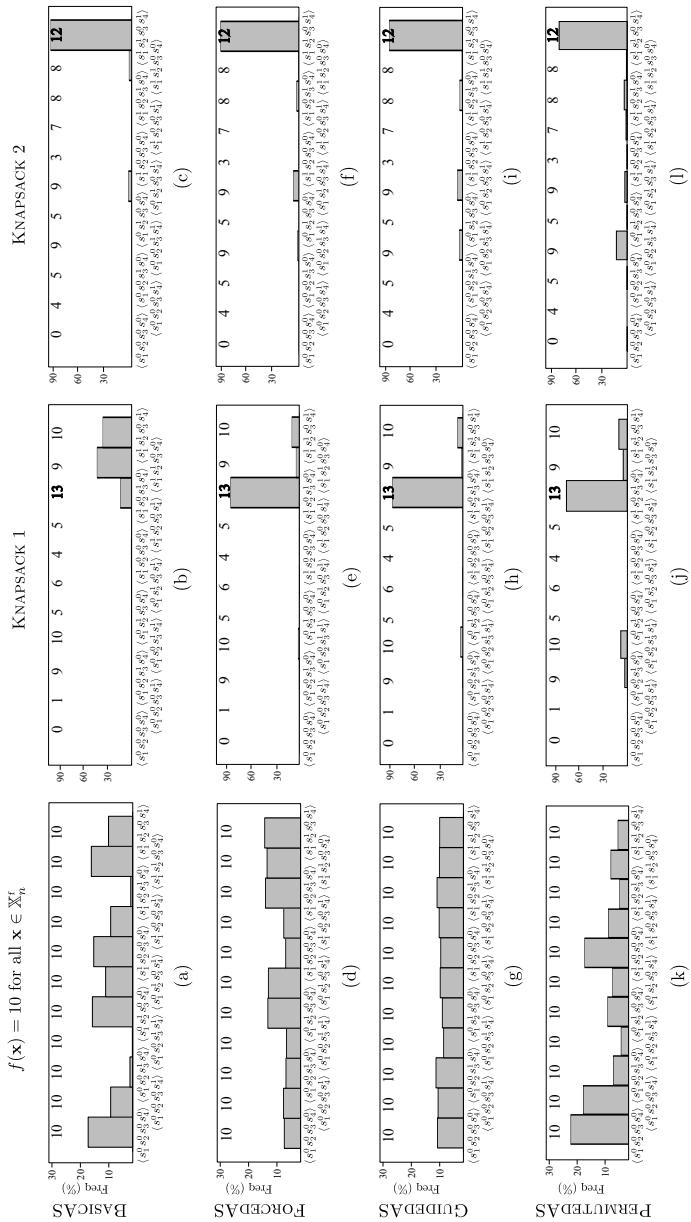


Figure 5.7: Frequency of convergence (based on 1000 runs) of the four AS variants (from top to bottom: BasicAS, ForcedAS, GuidedAS and PermutedAS) to all feasible solution sequences depicted in Figure 5.4; (a, d, g, k): in case of equal objective function values for all feasible solutions. (b, e, h, j): in case of knapsack problem 1 (see Table 5.2). (c, f, i, l): in case of knapsack problem 2. The numbers above the bars represent the objective function values of the corresponding solutions.

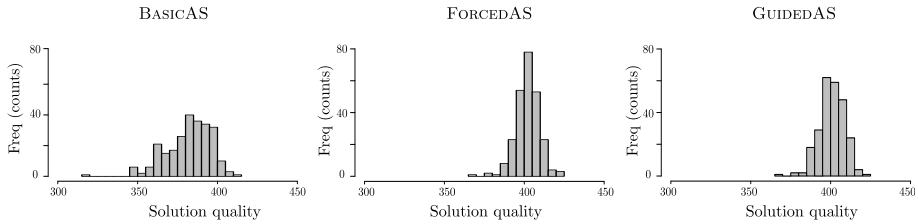


Figure 5.8: Frequency counts for a knapsack problem containing 60 items (generated according to [84]). The frequency counts are based on the objective function values of the solutions to which the algorithms converged. These frequency counts are based on 250 runs of each AS variant.

the quality of the solutions that are found by AS. As an illustration, consider Figure 5.8. For each AS variant, this figure gives an (empirical) distribution of the objective function values of the solutions that are found. For GuidedAS, the construction tree heuristic was used. For each AS variant, the initial pheromone concentration was 5000, $\rho = 0.3$ and we chose $K = 10$ ants. Here as well, an individual run was considered to have converged as soon as a particular solution occurred at least 18 times in a series of 20 consecutive solution sequences. It can be seen that both modified algorithms outperform the basic algorithm. However, ForcedAS and GuidedAS lead to solutions with similar objective function values. It should be noted that the number of iterations needed to achieve convergence was approximately the same for all variants (on average 380 generations for the depicted problem). As such, the number of times the fitness function f had to be evaluated before achieving convergence differed little between the three variants.

5.5.4. Artificial problems

Experiments with knapsack problems indicate that BasicAS is often outperformed by the adapted algorithms. The difference between ForcedAS and GuidedAS is less clear (Figure 5.8). Although GuidedAS is assumed to be less biased, it is not able to outperform ForcedAS for knapsack problems. However, when considering the structure of a knapsack problem, it can be seen that optimal solutions are positioned at the boundary of the feasible space. As can be derived from the construction tree and the results in Figure 5.7 (b), the ForcedAS variant focuses on these regions. This internal property of ForcedAS might explain this behavior. To verify this hypothesis, artificial subset selection problems (S, f, g) were created, in which the link between the objective function value of a solution and its closeness to the infeasible region is broken. In the following procedure, $d_H(\mathbf{x}, \mathbf{x}')$ denotes the Hamming distance between the n-tuples $\mathbf{x}$ and $\mathbf{x}'$.

1. Define a set of items S and a search space $X_{n]}$.
2. Compute the objective function value for each $\mathbf{x} \in X_{n]}$ as follows:

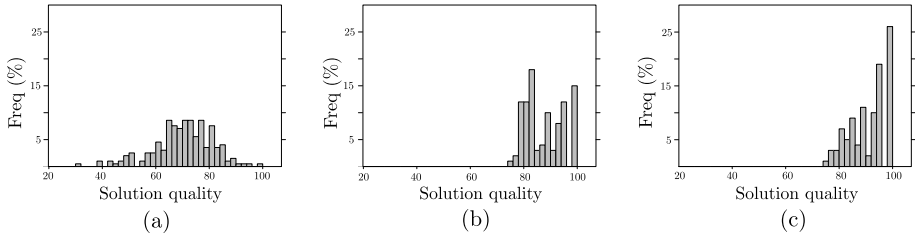


Figure 5.9: (a) Frequency distribution of all objective function values present in this artificial problem. (b) Frequency counts of the objective function values of the solutions to which ForcedAS converged. (c) Frequency counts of the objective function values of the solutions to which GuidedAS converged. These counts are the result of 100 individual runs of both AS variants.

(i) Choose a set of prototypes $F = \{\mathbf{x}_1^*, \ldots, \mathbf{x}_k^*\} \subset X_{n]}$ and assign an objective function value to each prototype: $f(\mathbf{x}_1^*), \ldots, f(\mathbf{x}_k^*)$.

(ii) For each $\mathbf{x} \in X_{n]} \setminus F$, set $f(\mathbf{x}) = \frac{\sum_{\{\mathbf{x}_i^* \in F\}} d_H(\mathbf{x}, \mathbf{x}_i^*)^{-1} f(\mathbf{x}_i^*)}{\sum_{\{\mathbf{x}_i^* \in F\}} d_H(\mathbf{x}, \mathbf{x}_i^*)^{-1}}$.

3. Compute the feasibility for each $\mathbf{x} \in X_{n]}$ as follows:

(i) Choose a set of prototypes $G = \{\mathbf{x}_1^+, \ldots, \mathbf{x}_\ell^+\} \subset X_{n]}$ and assign a feasibility value (0 or 1) to each prototype: $g(\mathbf{x}_1^+), \ldots, g(\mathbf{x}_\ell^+)$.

(ii) For each $\mathbf{x} \in X_{n]} \setminus G$, set $g(\mathbf{x}) = 1$ if $\frac{\sum_{\{\mathbf{x}_i^+ \in G\}} d_H(\mathbf{x}, \mathbf{x}_i^+)^{-1} g(\mathbf{x}_i^+)}{\sum_{\{\mathbf{x}_i^+ \in G\}} d_H(\mathbf{x}, \mathbf{x}_i^+)^{-1}} > 0.5$.

The objective function f, that is obtained by applying the procedure above, interpolates the objective function values of a predefined set of prototypes. Similarly, the feasibility function g interpolates the feasibilities of a (different) set of prototypes. Since both sets are chosen independently, the objective function value of a solution is independent of the distance to the feasibility boundary.

Using the procedure described above, several artificial subset selection problems were generated and used as a test bench for ForcedAS and GuidedAS. Figure 5.9 shows the distribution of the objective function values of the solutions that were found by both AS variants for a representative example with 16 items (initial pheromone concentration, number of ants, evaporation rate and convergence criterion were chosen as in the previous experiment). From this figure, it can be seen that the solutions found by GuidedAS have (on average) higher objective function values than the solutions found by ForcedAS. As such, we can conclude that GuidedAS to is able to outperform ForcedAS in this setting.

5.6. Max-Min Ant System with bias-avoidance

Up to this point, the central theme of this chapter was the study of bias in AS and how it can be avoided or mitigated. The experiments (on artificial problems) that were presented illustrate that the incorporation of our bias mitigating strategies leads to a procedure that outperforms the original AS. However, the development of top-performing procedures typically relies on the fruitful combination of multiple algorithmic ideas, and not only the bias-avoidance strategies presented here. Therefore, we opted to combine existing ideas, such as for example the incorporation of elitism, with our bias avoidance strategies in an attempt to develop a top-performing procedure. To that end, the Max-Min Ant System (MMAS) can be seen as one of the most efficient ACO procedures developed up to now. In essence, MMAS combines two ideas (elitism and pheromone trail constraining) that lead to an efficient procedure.

Bias-avoidance mechanisms can be easily incorporated into MMAS. Our implementation closely follows the framework described in [62] (note that we consider a maximization problem rather than a minimization problem in [62]). More precisely, let $\mathbf{x}^{\text{GB}}$ be the globally best solution found up to generation t. We define the pheromone trail limits $\tau_{\text{MAX}} = \frac{1}{\rho} f(\mathbf{x}^{\text{GB}})$ and $\tau_{\text{MIN}} = \tau_{\text{MAX}}(1 - \sqrt[n]{p_{\text{best}}})/\sqrt[n]{p_{\text{best}}}$, where $p_{\text{best}} \in [0, 1]$ is an additional parameter that is related to the probability that the best solution so far will be constructed. The pheromone trail constraining procedure uses these bounds as described hereafter. The pheromones are constrained to lie within the interval $[\tau_{\text{MIN}}, \tau_{\text{MAX}}]$ on line 11 of the procedure, *i.e.* when a pheromone trail parameter has a value smaller (resp. larger) than τ_{MIN} (resp. τ_{MAX}), it is set equal to τ_{MIN} (resp. τ_{MAX}). Below, we provide the pseudocode of the implementation that was used for the experiments with GUIDEDMMAS.

```
 1: procedure GUIDEDMMAS(X_n], f, g)
 2:     T(0) ←InitializePheromones
 3:     for all generations do
 4:         solutionsLastGen ← NULL
 5:         for ant = 1, ..., K do                                  ▷ make paths
 6:             x ← construct solution using decision rule (5.11)
 7:             solutionsLastGen.add(x)
 8:         end for
 9:         x^IB ← solution sequence in 'solutionsLastGen' with highest fitness
10:         Use only x^IB to update pheromones using guidedUpdate
11:         Update x^GB, τ_MAX and τ_MIN and keep pheromones within limits
12:     end for
13:     return x^GB
14: end procedure
```

In this chapter, we do not present any experiments with this GUIDEDMMAS. The

reason for this is twofold. Firstly, to be able to answer the question whether GUIDEDMMAS can compete with the state of the art in a wide variety of subset selection problems requires an extensive simulation study (including well-thought problem instances, fair comparison measures, etc.). Such a study is outside the scope of this dissertation and can be a topic of further research. Secondly, in the following chapter, we will use GUIDEDMMAS to solve a sample subset selection problem that uses a compositional representation of the samples. That chapter will include experiments where GUIDEDMMAS is compared with alternative approaches.

5.7. The sample subset selection problem

We now return to the original problem of selecting a subset of a given size k from a collection of mixtures. When both $|S|$ and k are large (in the following chapter we have that $n \approx 1000$ and $k = 100$) a brute force application of GUIDEDAS would require the construction tree heuristic. However, the constraint used in this problem allows an efficient computation of (5.11) and (5.12) without the need of an explicit form of the construction tree. It can easily be seen that for a given partial solution $\mathbf{y}^i$, it holds that

$$\left|L^{\mathrm{f}}_{\mathbf{y}^i}\right| = \binom{k - |\mathbf{y}^i|}{n - i}.$$

Therefore, GUIDEDAS can be applied very efficiently here.

The observation above suggests that GUIDEDAS is particularly useful for optimizing the score functions described in the first chapter of this part of the dissertation. Consequently, this procedure will be used in the next chapter as a heuristic for finding an optimal subset of a collection of mixtures.

5.8. Conclusions and discussion

The presence of (negative) search bias can have a strong influence on the performance of metaheuristics. This bias can be negative when it directs the search towards low-quality regions in the search space. This phenomenon has been studied intensively in the field of GAs. However, few studies deal with this bias in an ACO setting. In this chapter, the problem of negative search bias was studied and basic ideas were proposed to counter this bias. In several experiments, it was shown that (simple) bias-reducing techniques can improve the performance of AS for subset selection problems. In this chapter, we focused on the avoidance of bias to improve the performance of AS. However, the addition of an intentional bias towards promising regions might be worth considering as well. Moreover, the

inclusion of local search procedures may be another path that can be followed to obtain a better performing procedure. Nevertheless, it remains to be seen how local search procedures can be merged with the bias-avoidance strategies that were presented in this chapter.

The findings in Section 5.7 show that GuidedAS can be used efficiently to solve sample subset selection problems (Objective II.2). Moreover, the results that were obtained from experiments on synthetic problems hint that GuidedAS can be a promising procedure for finding good sample subsets.

The main theoretical finding is presented in Proposition 5.2, which states that the expected iteration quality of ForcedAS is a non-decreasing function of time. Even though the evolution of the expected iteration quality can be thought of as an interesting measure to assess the performance of an optimization procedure, other measures may be worth considering here as well. Therefore, future research may be directed towards finding new ways to quantify the behavior of ACO procedures.

We end this chapter by pointing out the tentative link between AS and Markov chain Monte Carlo (MCMC) methods [85]. This link becomes more apparent when we think about the pheromone matrix as a way to encode a probability mass function over all feasible subsets. The solutions that are constructed are observations drawn from that probability mass function. Subsequently, these observations are used to modify the parametrization (*i.e.* the pheromone matrix) of that distribution. Indeed, MCMC algorithms such as the Metropolis-Hastings [86] algorithm use a parametrized 'proposal' density function. The observations that are drawn from the proposal density function are used to change the values of its parameters. Interestingly, MCMC algorithms have been studied for several decades, and, as a result, are quite well understood. Due to the link between MCMC and AS, some of the results used in the MCMC field may be useful for gaining insight into AS. However, it remains to be seen whether the connection between MCMC and AS is strong enough for that.

6 Selecting an optimal subset of mixtures: numerical experiments

6.1. Introduction

In Chapter 4, the problem of selecting a subset of a collection of mixtures was translated into several formal optimization problems. Moreover, the resulting problems were classified as subset selection problems. Additionally, in Chapter 5, an Ant Colony Optimization procedure was proposed as a general heuristic for solving subset selection problems. In this chapter, the results from Chapters 4 and 5 will be used to select a subset of a collection of mixtures in a real-life application.

The chosen application is a data reduction problem in the field of agriculture. It should be noted that the application itself was proposed and described in detail in [41]. However, in that work, the compositional nature of the data was not taken into account. Moreover, based on the discussion in Chapter 4, the form of the score function used in [41] has several drawbacks. Finally, in [41] different heuristics were used to solve the subset selection problem. Therefore, this chapter can be seen as an extension of the work in [41]. Interestingly, the results obtained in [41] can be used as a reference to test the performance of the ACO procedure that was proposed in Chapter 5.

The remainder of this chapter is organized as follows:

- In Section 6.2, a real-life application in agriculture is introduced (original setting of [41]).
- In Section 6.3, the results obtained with the ACO procedures described in Chapter 5 are compared with the results obtained in [41].
- In Section 6.4, the compositional nature of the data is taken into account by using the score functions introduced in Chapter 4. In this section, the experiments focus on the influence of the metric as well as the type of metric-based score function that is used.

6.2. A subset selection problem in agriculture

6.2.1. General problem setting

The complete dataset in this study contains the representations of $n = 1033$ milk samples. Each sample is represented by a 45-part composition, that contains the (observed) proportional amounts (mass percentages) of $q = 45$ milk fatty acids. These milk samples belong to $e = 6$ different experiments, in which cows were subjected to different diets.

Milk fatty acids have been shown the potential to monitor dietary changes and diagnose metabolic disorders such as acidosis and ketosis [87]. The concentrations of the 45 fatty acids of the 1033 milk samples in the initial dataset were obtained using a simple but relatively inaccurate reference method (according to [88]). However, this simplified reference method is not able to resolve all milk fatty acids. To be able to study the fatty acid concentrations in detail the fatty acid concentrations need to be determined in a complete and accurate manner. However, for budgetary reasons one has to select a subset of $k = 100$ reference samples on which detailed milk fatty acids analysis is to be performed. The objective is thus to select a subset that is as informative as the total dataset, meaning that the variability of original dataset should be preserved in the subset.

Interestingly, this brief description fits relatively well within the discussion in Section 2.2.1. Indeed, there are no formal research goals presented that can be directly translated into an application-oriented score function. Therefore, we can apply a general purpose score function.

6.2.2. Problem representation and objective function

The problem that was described in the previous section is now written using the notation of Chapters 4 and 5.

The bit string representation (BSR) for this sample selection problem can be easily obtained. The item set S consists of the set of (1033) milk samples $(s_1, \ldots, s_{1033})$. Firstly, samples were ordered according to their experiment number, and within experiments the ordering was randomized. Subsequently, using that ordering, a search space X_n (with $n = 1033$) is defined. Pheromone trail parameters can be defined as described in Section 5.3.1. The objective function that will be used is the one proposed for GAs in [41]. Naturally, this objective function uses the representation of the samples by means of the mass percentages of the fatty acids. In words, the objective is to maximize the variance (of mass percentages) of the selected subset with the soft constraint that samples should be selected from each experiment. The size of the subset is constrained to be exactly 100. Note that

the requirement that a subset should contain at least two samples (otherwise the variance is zero) from each of the six experiments is not encoded explicitly as a problem constraint. However, solutions that do not respect this constraint are given an objective function value of zero (therefore, we call it a soft problem constraint).

The vectors of mass percentages are grouped into a 1033×45 matrix $\mathbf{Z}$. Moreover, let $\mathbf{Z}^j$ denote the matrix containing the mass percentages of the samples from the jth experiment ($j = 1, \ldots, e$). Following the notation introduced in Chapter 4, $(\mathbf{Z}^{\mathbf{x}})^j$ denotes the matrix that only contains the rows that correspond to samples that are present in subset $\mathbf{x}$ and are part of the jth experiment. Lastly, $(\mathbf{Z}^{\mathbf{x}})^j_{.,i}$ is the ith column of that matrix. Using this notation, the objective function is written as:

$$f(\mathbf{x}) = \min_{j=1}^{e} \left(\min_{i=1}^{q} \frac{\operatorname{var}\left((\mathbf{Z}^{\mathbf{x}})^j_{.,i} \right)}{\operatorname{var}\left((\mathbf{Z})^j_{.,i} \right)} \right) , \tag{6.1}$$

The required size k of the subset is encoded as a hard constraint implying $g(\mathbf{x}) = 1$ if $|\mathbf{x}| = k$ and $g(\mathbf{x}) = 0$ if $|\mathbf{x}| \neq k$. These functions are combined in the following mathematical optimization problem:

$$\begin{aligned} \underset{\mathbf{x} \in 2^S}{\text{maximize}} \quad & \min_{j=1}^{e} \left(\min_{i=1}^{q} \frac{\operatorname{var}\left((\mathbf{Z}^{\mathbf{x}})^j_{.,i} \right)}{\operatorname{var}\left((\mathbf{Z})^j_{.,i} \right)} \right) \\ \text{subject to} \quad & g(\mathbf{x}) = 0 \,, \end{aligned}$$

Since the number of items is large, a brute force application of GuidedAS would require the construction tree heuristic. However, the constraint used in this problem allows an efficient computation of (5.11) and (5.12) without the need of an explicit form of the construction tree. Indeed, as mentioned in the previous chapter, for a given partial solution $\mathbf{y}^i$, we have that

$$\left| \mathcal{L}^{\mathrm{f}}_{\mathbf{y}^i} \right| = \binom{q - |\mathbf{y}^i|}{n - i} .$$

6.3. Experiments and results

As the problem described before is a subset selection problem, the AS variants can be directly applied as described in the experimental section of Chapter 5. However, unlike in Chapter 5, where we illustrate that bias avoidance allows the

search procedure to focus on high-quality regions in the search space, we want to illustrate that the use of bias-avoidance mechanisms is beneficial in a practical application. As such, the experiments in this section differ from the experiments described before. Firstly, we implement our bias-avoidance mechanisms within the Max-Min Ant System (MMAS) [62]; as this younger AS variant often outperforms the original AS algorithm, it is likely to be a suited basis for our bias-avoidance strategies. The algorithm that is obtained by incorporating the bias-avoidance mechanisms in MMAS is further referred to as GuidedMMAS. Secondly, our goal will be to find a good solution sequence as efficiently as possible. As such, instead of desiring convergence, we aim to find a high-quality solution using only a limited number of iterations.

To be able to make a fair comparison between the existing evolutionary approach using GAs, the standard AS and MMAS variants, and the variants that incorporate bias-avoidance mechanisms, the total number of fitness function evaluations was fixed at 15000. Each algorithm was run 30 times, and per run the best solution with the highest fitness was stored. Table 6.1 reports the best, worst and average solution qualities (fitnesses) that were found over 30 runs for each algorithm, as well as the results for the non-evolutionary approaches reported in [41] for the same problem (note that best, worst and average solution qualities are equal here since these methods are non-stochastic). Moreover, for each algorithm, parameters were optimized by a grid search.

Table 6.1 shows that that the non-deterministic approaches are outperformed by GAs, GuidedAS and GuidedMMAS. Moreover, BasicAS and BasicMMAS are completely useless in this setting as they both seem unable to create a subset that contains items from all experiments (the soft constraint). This can be expected as the BSR that is used here will heavily suffer from construction bias. This construction bias forces these algorithms towards regions in the search space that violate the soft constraints (typically regions that will be favored by construction bias only contain experiments from the first two experiments). Even though GuidedAS outperforms the non-evolutionary algorithms, GAs are clearly performing better. We can nevertheless conclude that the adapted AS algorithm has an added value with respect to the original AS algorithm. The most competitive algorithm in this experiment is definitely GuidedMMAS. This application clearly illustrates that the implementation of bias-avoidance mechanisms into MMAS can lead to a well-performing algorithm.

Finally, we devote some attention to PermutedAS. From the results in Table 6.1 it is clear that PermutedAS does not perform well in this setting. Moreover, the performance results obtained with PermutedAS are not better than the ones obtained by a random (undirected) search through the search space (i.e. randomly picking 15000 feasible points and selecting the one with the highest fitness). Additional experiments with PermutedAS showed that it never converged

Table 6.1: Summary of the best, worst and average solution qualities found by eight subset selection methods on the application in agriculture (results based on 30 runs per algorithm). The results of the Kennard and Stone algorithm are presented with the Euclidean Distance (ED). For the AS algorithm the results of Basic(MM)AS, PermutedAS, GuidedAS and GuidedMMAS are listed.

Method	Best quality	Worst Quality	Average Quality
ED	0.400	0.400	0.400
k-means	0.089	0.089	0.089
OptiSim	0.100	0.100	0.100
GA	0.880	0.679	0.787
Basic(MM)AS	0.000	0.000	0.000
PermutedAS	0.415	0.296	0.354
GuidedAS	0.743	0.593	0.662
GuidedMMAS	0.946	0.694	0.807

(after 14 days, computations were stopped). The strong influence of the assignment order on the probability of constructing a particular solution sequence can explain this behavior. Indeed, when $\tau_i^0 = \tau_i^1$ for $j = 1, \ldots, n$, for a given ordering, the probability that, for example, the set of selected items will be a subset of the first 200 items is approximately $2^{832} \approx 10^{250}$ times the probability that this selection is a subset of the last 200 items. As such, even in the (unlikely) case that the pheromone matrix is close to convergence, the constantly changing assignment orderings will prevent it from achieving complete convergence, and the result will be a pseudo-random search through the search space.

6.4. Metric-based score functions in practice

From the discussions in Chapter 5, it follows that objective function given in Eq. (6.1) has several (potential) drawbacks:

- As Eq. (6.1) maximizes the marginal variances, it will strongly focus on extreme samples, *i.e.* samples that have a representation that lies close to the convex hull of the dataset.
- As Eq. (6.1) maximizes the marginal variances, it does not take the compositional nature of the data into account.

To overcome these problems, the metric-based score functions introduced in Chapter 5 will be used here. Moreover, from a methodological point of view, we did not feel the need here to incorporate the experiment number into the objective nor in the constraints. We experiment with the following objective functions:

$$f_1(\mathbf{x}) = \min_{i \neq j} \left(d\left(\mathbf{Z}^{\mathbf{x}}_{i,.}, \mathbf{Z}^{\mathbf{x}}_{j,.}\right)^2 \right), \tag{6.2}$$

$$f_2(\mathbf{x}) = \left(\frac{1}{100} \sum_j \left(\min_i \left(d\left(\mathbf{Z}^{\mathbf{x}}_{i,.}, \mathbf{Z}^{-\mathbf{x}}_{j,.}\right)^2 \right) \right) \right)^{-1}, \tag{6.3}$$

$$f_3(\mathbf{x}) = \left(\max_j \left(\min_i \left(d\left(\mathbf{Z}^{\mathbf{x}}_{i,.}, \mathbf{Z}^{-\mathbf{x}}_{j,.}\right)^2 \right) \right) \right)^{-1}. \tag{6.4}$$

In these objective functions, the metric that is used is either Euclidean distance (denoted f_1^{e}, f_2^{e} and f_3^{e}) or the Aitchison distance (denoted f_1^{a}, f_2^{a} and f_3^{a}). Note that the exponents in (6.3) and (6.4) were used to obtain score functions that need to be maximized.

6.4.1. Aitchison distance versus Euclidean distance

From the discussion in the introductory chapter on compositional data, it could be concluded that (from a methodological point of view) the Aitchison distance is to be preferred over Euclidean distance when composing a metric based score function. Moreover, in Chapter 4, it was illustrated with an artificial problem that the metric that is used strongly influences the types of subsets that are selected. However, from a more pragmatic point of view, the question may be raised whether this observation translates to real-life datasets. To look into this problem, a small experiment was set up. The results are presented in Table 6.2. The experimental setup is summarized hereafter:

1. Fatty acids for which at least one sample with a concentration of zero was reported, were removed from the dataset. This action can be justified by the fact that the FA concentrations were measured by means of an inaccurate reference method. More extensive measurements showed that the zeros that are present in this dataset are rounding zeros. Nevertheless, we recognize that the deletion of these FAs can have an impact on the subset that is selected. In total, 17 FAs were retained.

2. f_1^{e} was maximized using GuidedAS. The optimization procedure was run 5 times.

3. The (five) subsets that resulted from the previous step were scored using f_1^{e} and f_1^{a} (separately).

4. The minimum and the maximum of the scores computed using f_1^{e} and f_1^{a} are presented in Table 6.2.

5. Steps (2)–(4) are repeated for score functions f_2^{e} and f_2^{a}, and f_3^{e} and f_3^{a}.

From the results that are presented in Table 6.2, it can be suspected that the metric (Euclidean versus Aitchison) that is used to build a metric-based score function has a strong influence on the subsets that will be selected. More precisely, a subset with a high score for a score function that uses the Euclidean distance may not be optimal with respect to a score function that uses the Aitchison distance. For example, consider columns f_1^{e} and f_2^{a} in Table 6.2. From the five runs of GuidedAS in which f_1^{e} was optimized, the best subset (according to score function f_1^{e}) had a score of 6.32×10^{-4}. On the other hand, from the five runs of GuidedAS in which f_1^{a} was optimized, the best subset (according to score function f_1^{e}) had a score of 1.18×10^{-4}. Interestingly, when comparing this last number with the values in column f_1 (the random case), it can be seen that the best of 100 randomly drawn subsets had a score of 1.66×10^{-4}, which is clearly higher than 1.18×10^{-4}. Similar observations can be made for the other score functions. Lastly, it should be mentioned that the magnitudes of the similarities or dissimilarities of the scores that are observed in this experiment possibly depend on the characteristics of the dataset at hand. However, as this experiment illustrates, one cannot blindly say that the influence of the loss function that is choosen can be neglected beforehand.

6.4.2. Maximizing diversity or representability

Recall from the first chapter of this part, that we distinguished between two types of metric-based score functions. The first type exists of score functions can be used to find a set with maximal variability, whereas the second type exists of score functions that can be used to find a subset that is representable for a given dataset. It is clear that f_1 belongs to the former type whereas f_2 and f_3 belong to the second type. As in the previous subsection, it can be asked whether these score functions are interchangeable in practical situations. To look into this question, an experiment was setup. The experimental setup is described hereafter.

1. Fatty acids for which at least one sample with a concentration of zero was reported, were removed from the dataset.
2. f_1^{a} was maximized using GuidedAS. The optimization procedure is run 5 times.
3. For each run and each score function (f_1^{a}, f_2^{a} and f_3^{a}) the subset (from those five) with the lowest (resp. highest) score is reported in Table 6.3.
4. Steps (2)–(3) are repeated for f_2^{a} and f_3^{a}.

Table 6.3 shows the performance results obtained in this experiment. As a reference, the last column reports the scores of 100 randomly generated subsets. Firstly, as expected, it can be concluded that the explicit maximization of a specific score

Table 6.2: Performance results for different score functions. Per column, one objective function was maximized (5 times), the rows present the minimum and maximum scores observed using Euclidean distance or Aitchision distance. The last three columns present the minimal and maximal scores for a sequence of 100 randomly generated subsets. The numbers in these columns can be used as a reference.

Performance metric			Objective function								
			f_1		f_2		f_3		Random		
			f_1^{e}	f_1^{a}	f_2^{e}	f_2^{a}	f_3^{e}	f_3^{a}	f_1	f_2	f_3
Euclidean f^{e}	min		5.54×10^{-4}	6.32×10^{-5}	2.47×10^{3}	1.88×10^{3}	5.41×10^{2}	1.61×10^{2}	6.42×10^{-7}	1.16×10^{3}	4.53×10^{1}
	max		6.32×10^{-4}	1.18×10^{-4}	3.54×10^{3}	1.98×10^{3}	6.64×10^{2}	3.16×10^{2}	1.66×10^{-4}	1.85×10^{3}	2.86×10^{2}
Aitchision f^{a}	min		7.21×10^{-2}	3.64×10^{-1}	2.46	3.32	2.33×10^{-1}	8.82×10^{-1}	2.10×10^{-2}	1.69	4.23×10^{-2}
	max		1.32×10^{-1}	5.01×10^{-1}	2.54	3.39	3.83×10^{-1}	1.05	1.34×10^{-1}	2.56	4.12×10^{-1}

Table 6.3: Performance results for different score functions. Per column, one objective function was maximized (5 times), the rows present the minimum and maximum scores observed using either f_1^{a}, f_2^{a} or f_3^{a}. The last three columns present the minimal and maximal scores for a sequence of 100 randomly generated subsets. The numbers in these columns can be used as a reference.

			Objective function			
			f_1	f_2	f_3	Random
Performance function	f_1	min	0.35	0.07	0.12	0.02
		max	0.50	0.12	0.17	0.13
	f_2	min	2.33	3.32	3.25	1.69
		max	2.53	3.39	3.31	2.56
	f_3	min	0.05	0.63	0.88	0.04
		max	0.38	0.87	1.05	0.41

function leads to a subset that has a high score for the score function that was optimized. Moreover, as compared to the randomly generated subsets, there is a clear increase in the scores that are obtained. However, a subset that results from optimizing $f_2^{\mathbf{a}}$ will generally have a low score for $f_1^{\mathbf{a}}$. Surprisingly, the best of 100 randomly generated subsets had a higher score for $f_1^{\mathbf{a}}$ than a subset that results from optimizing $f_2^{\mathbf{a}}$. On the other hand, a subset that results from optimizing $f_3^{\mathbf{a}}$ will generally have a high score for $f_2^{\mathbf{a}}$ (a similar conclusion can be drawn when $f_2^{\mathbf{a}}$ and $f_3^{\mathbf{a}}$ are interchanged). Therefore, from these experiments we can cautiously conclude that the tradeoff between the maximization of variability and representativity persists in this problem. Subsets with high scores for $f_1^{\mathbf{a}}$ (a score function that favors variability) do not necessarily have high scores for $f_2^{\mathbf{a}}$ and $f_3^{\mathbf{a}}$ (score functions that favor representativity), or vice versa. On the other hand, subsets that are obtained by maximizing $f_2^{\mathbf{a}}$ with GuidedAS generally have high scores for $f_3^{\mathbf{a}}$, or vice versa.

6.5. Conclusions and discussion

The main objective of this chapter was the performance evaluation of the ACO procedure that was derived in the previous chapter in a real-life setting (Objective II.3). Moreover, the metric-based score functions that were described in Chapter 4 were critically evaluated in a real-life setting.

6.5.1. Performance of ACO procedures

The experimental results presented in Section 6.3 illustrated that GuidedMMAS performs well in practice. From the results that are reported, it seems that GuidedMMAS is competitive and can potentially outperform competing methodologies,

such as GAs in particular. Moreover, the combination of bias-avoidance techniques with MMAS allows to convert the basic Ant System from an (in this setting) useless approach into a well-performing procedure.

Even though the results that were presented in this chapter show that GuidedMMAS (or perhaps bias-avoidance in general) is a well-performing procedure, several remarks can be made with respect to the approach that is presented here, or the experimental setup in general. The following remarks can be interpreted as personal criticism and suggestions for further research. Firstly, it can be asked whether the comparison between the subset selection approaches that is presented here is fair. For example, to compare the AS procedures with GAs in Table 6.1, we choose to limit the number of function evaluations that is allowed. To investigate the potential of an optimization procedure, such an approach can be considered as being reasonable as the evaluation of the objective can be computationally most demanding. On the other hand, several problems have an objective function that can be evaluated efficiently (for example, Eq. (6.1) can be evaluated rather efficiently). In those cases, the constructive manner in which an AS procedure generates new candidate solutions is computationally demanding in some cases. Indeed, to construct a new candidate solution, a large number of pseudo-random numbers need to be generated, and especially for GuidedMMAS the computation of $|\mathcal{L}^{\mathrm{f}}_{\mathbf{y}^i}|$ can be demanding (even in the simple case presented here). In those cases, fixing the actual run-time might be more fair than fixing the number of function evaluations. However, such an approach will put an additional emphasis on an efficient implementation of the procedures. This raises the question on how GuidedMMAS can be optimized to obtain an optimal run-time efficiency.

6.5.2. An evaluation of metric-based score functions

The experiments in Subsection 6.4.1 suggest that, for a given metric-based score function, the replacement of the Euclidean distance with the Aitchison distance has a strong impact on the subsets that are selected. We conclude that, besides its mathematical elegance, the Aitchison distance also influences the type of subsets that are selected in a real-life subset selection problem.

Finally, from the results in Subsection 6.4.1, we conclude that there exists a tradeoff between subsets that are diverse and subsets that are representative. However, it must be recognized that these conclusions are based on a single real-life test case. It remains to be seen whether these conclusions translate to other subset selection settings. Moreover, the results presented here are based on five runs per case (of a stochastic procedure). Consequently, the results that are obtained should be interpreted with some caution.

PART III

A SET ESTIMATOR FOR THE UNMIXING OF MIXTURES

7 Unmixing of a mixture with a set-based representation of the sources

7.1. Introduction

We start this introduction with an example of an *unmixing problem*.

Assume you have been given a cup of water with a salinity of 21 ppt (parts per thousand). Moreover, you are told that the water in this cup is a blend of fresh water (salinity of 2 ppt) and sea water (salinity of 40 ppt). Subsequently, you are asked to estimate the proportional amount of fresh water (x_1) in the cup.

It may seem rather intuitive to give $x_1 = 50\%$ as an answer to this problem. Indeed, when the proportional amount reflects the ratio of the number of fresh water molecules to the total number of molecules in the cup, this assertion is valid. Using the terminology of the chapter on compositional data analysis (Chapter 2), the salinity can be used to represent the mixture ($\mathbf{y}$), the source of fresh water ($\mathbf{c}_1 = 2$) and the source of sea water ($\mathbf{c}_2 = 40$). To obtain the answer $x_1 = 50\%$, we can solve the following system of linear equations.

$$\begin{cases} x_1\,\mathbf{c}_1 + x_2\,\mathbf{c}_2 &= \mathbf{y} \\ x_1 + x_2 &= 1 \end{cases}.$$

Naturally, the solution of this system should satisfy $x_1, x_2 \geq 0$. Even though this description may seem rather trivial at first sight, it entails one important assumption that is generally referred to as the linear mixing model (LMM) assumption. Loosely speaking, this assumption implies a linear relationship between the fractional abundance of the fresh water (x_1) in the mixture and the representation of this mixture (the salinity in this case). The first equation in the system above expresses this assumption mathematically. The second equation states that x_1 and x_2 should be proportions (*i.e.* add up to one). In this example, this assumption may seem rather trivial. Unfortunately, this is not necessarily the case in all applications. We elaborate on this later.

We will generally refer to the fresh water and the sea water as *sources* and the vectors $\mathbf{c}_1$ and $\mathbf{c}_2$ as *prototype vectors* of these sources; $\mathbf{y}$ is called the *mixture vector*.

In this example, where the representations of the mixture and the sources are scalars, one can readily obtain an estimate for x_1 and x_2. However, in most

applications, the information about the sources is less explicit. For instance, we might only know that $\mathbf{c}_1 \in [1,3]$ and $\mathbf{c}_2 \in [30,45]$. In this setting, any solution $(x_1, x_2, \mathbf{c}_1, \mathbf{c}_2)$ to the system

$$\begin{cases} \mathbf{y} &= x_1\mathbf{c}_1 + x_2\mathbf{c}_2\,, \\ 1 &= x_1 + x_2\,, \\ x_1 &\geq 0\,, \\ x_2 &\geq 0\,, \\ \mathbf{c}_1 &\geq 1\,, \\ \mathbf{c}_2 &\geq 30\,, \\ \mathbf{c}_1 &\leq 3\,, \\ \mathbf{c}_2 &\leq 45\,, \end{cases}$$

can be considered to contain a possible estimate for x_1 and x_2. This means that the information on the sources lacks the required precision to allow for a unique x_1 and x_2 to be extracted.

We now briefly elaborate on the origin of this imprecision. Firstly, we state what it is not: the imprecision is not due to some measurement error, neither does it result from an underlying stochastic process. Instead, it reflects the imprecision that is inherent in the way the source is defined. Here, the sources are stated to be fresh and sea water. Naturally, there exists a variety of fresh and sea water sources. For example, the salinity of water taken from the Mediterranean Sea differs from the salinity of water taken from the North Sea. Therefore, a precise answer to the question raised before cannot be given. Instead, we can provide **a set** of values (proportions) for x_1 that can explain our observation. More precisely, for each proportion in this set, there exists at least one couple of prototype vectors $\mathbf{c}_1 \in [1,3]$ and $\mathbf{c}_2 \in [30,45]$ that could have been used to create a mixture such that $\mathbf{y} = x_1\,\mathbf{c}_1 + (1 - x_1)\,\mathbf{c}_2$.

For the example given above, it can easily be shown that the solution set for x_1 (and x_2) is a closed interval. The minimal (resp. maximal) element of this interval can be found by solving the following minimization (resp. maximization) problem:

$$\begin{aligned} \underset{(x_1,x_2,\mathbf{c}_1,\mathbf{c}_2)\in\mathbb{R}^4}{\text{minimize}} \quad & x_1 \\ \text{subject to} \quad & \mathbf{y} = x_1\,\mathbf{c}_1 + x_2\,\mathbf{c}_2\,, \\ & 1 = x_1 + x_2\,, \\ & x_1 \geq 0\,, x_2 \geq 0\,, \\ & \mathbf{c}_1 \in [1,3]\,, \\ & \mathbf{c}_2 \in [30,45]\,. \end{aligned}$$

In this toy example where $\mathbf{y}, \mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}$, there exists a simple, closed-form solution. However, the unmixing problem becomes non-trivial in the more general setting where $\mathbf{y}, \mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^q$ and $\mathbf{c}_1 \in [1, 3]$ and $\mathbf{c}_2 \in [30, 45]$ are replaced by $\mathbf{c}_1 \in C_1$ and $\mathbf{c}_2 \in C_2$ with C_1 and C_2 two subsets of $\mathbb{R}^q$. In the introductory example, such a situation may occur when we are provided with a more extensive characterization of the fresh and sea water. For example, we could use the characterization of water in terms of the concentrations of calcium, potassium and bromine[1]. Similarly, the unmixing problem becomes more complicated when more than two sources are considered. In the introductory example, this situation could occur when the water in the cup is a mixture of deep ocean water, sea water and fresh water.

In Part III of this dissertation we will develop a general methodology for solving unmixing problems such as the one presented in the introductory example. In the present chapter, we illustrate that these problems occur frequently in (applied) research and review related literature. In the following chapter, we develop a series of algorithms that can be used to solve the mathematical optimization problems that are encountered in these problem settings.

The remainder of this chapter is organized as follows:

- In Section 7.2, the unmixing problem that is described above is generalized and formalized.
- In Section 7.3, several problem settings are presented that allow to derive a set-based representation of sources from data.
- In Section 7.4, several (potential) applications are presented.
- In Section 7.5, related work is reviewed.
- In Section 7.6, conclusions are presented and discussed.

7.2. Formal problem description

A multivariate process can often be interpreted as a mixture of multiple (n) source processes. Several applications require an estimate of the proportional contribution of at least one of these sources to such a mixture. In this chapter, we propose a procedure that provides a set-based estimate of the proportional contribution of a source of interest to a mixture. To be able to provide such an estimate, we will –as most existing approaches do– require that the mixture and the sources are represented in the same Euclidean vector space. When such a vector representation is available, the linear mixing model (LMM) (see for instance [89]) is the most popular model to describe such a mixture. In its most general form, this model

[1] In this example, the concentrations of calcium, potassium and bromine are the *features* that are used to describe or characterize a mixture.

can be written as

$$\mathbf{y} = \sum_{i=1}^{n} x_i \, \mathbf{c}_i \, , \tag{7.1}$$

with prototype vectors $\mathbf{c}_1, \ldots, \mathbf{c}_n \in \mathbb{R}^q$, mixture vector $\mathbf{y} \in \mathbb{R}^q$ and vector of mixing coefficients $\mathbf{x} = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$ such that $\sum_{i=1}^n x_i = 1$ and $0 \leq x_i \leq 1$. In this model, x_i represents the proportional contribution of the ith source to the mixture.

When the prototype vectors are given, deriving $x_1, \ldots, x_n$ is trivial (we briefly return to the setting in the following chapter). In practice, however, prototype vectors are rarely known explicitly. Instead, we only have a rough description of these prototype vectors. In some cases, probability density functions can be used to model this lack of knowledge (either assumed to be given or estimated from data), and Bayesian modeling approaches are used to derive estimates for x_i (or credibility intervals if needed). However, often the amount of information (or data) that is available is insufficient to specify (or estimate) the required probability density functions. In such cases, one must resort to a less expressive way of describing the sources. One option consists of representing a source by a set, *i.e.* the ith source is represented by the set $C_i \subseteq \mathbb{R}^q$, representing the knowledge that the ith prototype vector $\mathbf{c}_i \in C_i$.

Now, let $C_1, \ldots, C_n \subseteq \mathbb{R}^q$ and $\mathbf{y} \in \mathbb{R}^q$ be given, the feasible set $X(\mathbf{y}, (C_i)_{i=1}^n)$ is defined as

$$X(\mathbf{y}, (C_i)_{i=1}^n) = \left\{ \mathbf{x} \in \mathbb{S}^n \mid \left(\exists \, (\mathbf{c}_i)_{i=1}^n \in \bigtimes_{i=1}^n C_i \right) \left(\mathbf{y} = \sum_{i=1}^n x_i \, \mathbf{c}_i \right) \right\} \, , \tag{7.2}$$

where $\mathbb{S}^n$ is the n-dimensional simplex. This means that each element in $X(\mathbf{y}, (C_i)_{i=1}^n)$ is a vector of mixing proportions that respects our knowledge about the sources and the LMM. Moreover, we define the following projection of $X(\mathbf{y}, (C_i)_{i=1}^n)$:

$$X^k(\mathbf{y}, (C_i)_{i=1}^n) = \{ z \in [0,1] \mid (\exists \, \mathbf{x} \in X(\mathbf{y}, (C_i)_{i=1}^n))(x_k = z) \} \, .$$

The set $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ can be interpreted as the set of all *possible* values for the proportional contribution of the kth source to the mixture represented by $\mathbf{y}$. As such, any element from $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ can be used as a point estimate for x_k. Moreover[2], without making further assumptions, one cannot select a single *best* element out of X^k. Because of that, we propose X^k as a set estimator for x_k. Fortunately, as we will show later, under quite general conditions we have that $X^k = [\inf(X^k), \sup(X^k)]$, consequently, in these cases we will refer to our estimator as an interval estimator. In those cases, X^k can be considered as a natural extension

[2] For notational convenience we will use X^k as the short-hand notation for $X^k(\mathbf{y}, (C_i)_{i=1}^n)$, the arguments $\mathbf{y}$ and $(C_i)_{i=1}^n$ can be dropped, as they often can be considered fixed. The extended notation will reappear in Section 5.

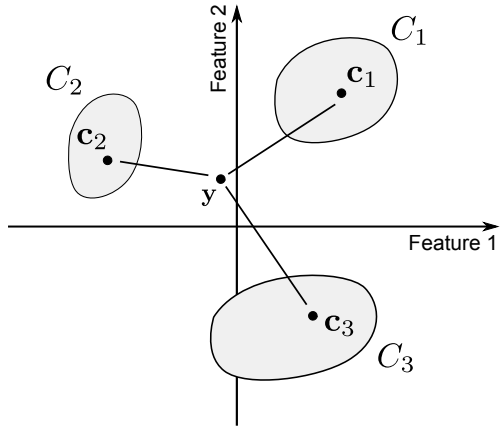


Figure 7.1: Graphical representation of the set-based description of three sources (C_1, C_2 and C_3) as well as three prototype vectors ($\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$) that were used to create a mixture with representation $\mathbf{y}$.

of the more familiar point estimator of x_k. Here, the computation of this interval can be performed by solving two optimization problems. This is the main topic of the next chapter. Interestingly, it will turn out that the resulting optimization problems can be solved globally in an efficient manner.

A graphical representation

The data ($\mathbf{y}$ and $C_1, \ldots, C_n$) can easily be visualized. Figure 7.1 illustrates the geometry of this problem in case $n = 3$ and $q = 2$.

Convex sets

We end this section by pointing at the special, yet interesting case where the sets $C_1, \ldots, C_n$ are compact and convex. We will show in the following chapter that in this case $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ is an interval, *i.e.* we have that $X^k(\mathbf{y}, (C_i)_{i=1}^n) = [\inf(X^k(\mathbf{y}, (C_i)_{i=1}^n)), \sup(X^k(\mathbf{y}, (C_i)_{i=1}^n))]$. This property is appealing, as we are used to expressing uncertainty on a predicted value by means of an interval. Moreover, to compute X^k, it suffices to find its minimal and its maximal elements. This naturally leads to the following optimization problem:

$$
\begin{array}{lll}
\underset{\mathbf{x}\in\mathbb{R}^n, (\mathbf{c}_i)_{i=1}^n \in \mathbb{R}^q}{\text{minimize}} & x_k & \\
\text{subject to} & \mathbf{y} = \sum_{i=1}^n x_i \, \mathbf{c}_i \,, & \\
& 1 = \sum_{i=1}^n x_i \,, & \\
& x_i \geq 0 \,, & \text{for } i = 1, \ldots, n \,, \\
& \mathbf{c}_i \in C_i \,, & \text{for } i = 1, \ldots, n \,.
\end{array}
$$

It can be seen that directly solving this optimization problem will be difficult due

to the bilinear equality constraint. However, in the next section, we propose an equivalent[3] optimization problem that can be solved efficiently.

7.3. A set-based representation of sources

We will consider problem settings in which the sources are represented by means of sets instead of points. Therefore, we say that the description of the sources is set-based (as opposed to the case were the sources are represented by single points). As a result, we can refer to our estimator as a set-based set estimator. Indeed, it uses set-based representations of the sources ánd the resulting estimate is a set.

To be able to use our set estimator $X^k(\mathbf{y}, (C_i)_{i=1}^n)$, its arguments need to be specified. In a practical setting, we will be given a noisy observation of $\mathbf{y}$ (denoted $\tilde{\mathbf{y}}$) instead of $\mathbf{y}$. Moreover, the set-based representation $(C_i)_{i=1}^n$ of the sources may not be directly available in most applications. Naturally, there exist cases in which these sets are known, as for instance in the introductory example. However, this will not be the case in general. Fortunately, there exist several manners to obtain estimates $(\hat{C}_i)_{i=1}^n$ of $(C_i)_{i=1}^n$ in an indirect manner. In conclusion, we often will be forced to use $X^k(\tilde{\mathbf{y}}, (\hat{C}_i)_{i=1}^n)$ as an estimate for $X^k(\mathbf{y}, (C_i)_{i=1}^n)$.

7.3.1. Convex hulls

Even though a set-based representation $(C_i)_{i=1}^n$ of the n sources may not be directly available in most applications, in some of these applications, data is available that captures the (natural) variability in these sources. In these cases, the convex hull of these data points can be used to obtain conservative estimates of $(C_i)_{i=1}^n$ as illustrated in Figure 7.2. To motivate this approach, let us return to the introductory problem setting. Assume that sea water and silt water are characterized by means of multiple components. Moreover, let two matrices $\mathbf{C}^1 \in \mathbb{R}^{m_1 \times q}$ and $\mathbf{C}^2 \in \mathbb{R}^{m_2 \times q}$ be given in which the rows represent observations of sea water (C_1) and fresh water (C_2). We refer to these observations as *observed prototypes*. Figure 7.2 shows a scatter plot of these datasets ($q = 2$) as well as their convex hulls (see Chapter 2 for a definition). Now let $\mathbf{a}$ and $\mathbf{b}$ be two elements of $\mathbf{C}^1$. Recall that $\mathbf{a}$ and $\mathbf{b}$ are representations of two types of sea water (*i.e.* two prototype vectors). These prototypes of sea water can be mixed to obtain a third prototype by choosing $\theta \in [0, 1]$ and defining $\mathbf{d} = \theta\,\mathbf{a} + (1 - \theta)\,\mathbf{b}$ (we typically say that $\mathbf{d}$ is a convex combination of $\mathbf{a}$ and $\mathbf{b}$). Naturally, a blend of two types of sea water results in a mixture that solely consists of sea water. Therefore, $\mathbf{d}$ represents sea water as well.

[3] Here equivalent means that the solution of the first can be readily obtained from the solution of the latter.

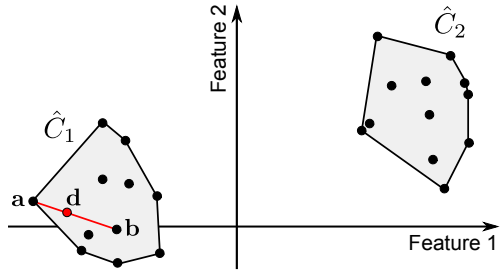


Figure 7.2: Scatter plots of two artificial datasets $\mathbf{C}^1$ (left) and $\mathbf{C}^2$ (right) as well as their convex hulls. The solid dots represent observed points, the red dot is a convex combination of $\mathbf{a}$ and $\mathbf{b}$. The red line segment groups all points that can be written as convex combinations of $\mathbf{a}$ and $\mathbf{b}$.

Interestingly, the convex hull of the observed prototype vectors in $\mathbf{C}^1$ (which is denoted $\text{conv}(\mathbf{C}^1)$) contains all vectors that can be constructed using this strategy. Therefore, it is a natural set to consider. This principle is illustrated in Figure 7.2. It should be noted that the resulting sets are generally not equal to C_1 and C_2, but only approximations of these sets. Therefore, we denote them as $\hat{C}_1$ and $\hat{C}_2$.

7.3.2. Extensions of the convex hull

Continuing along the reasoning above, the convex hulls are extremely conservative. Indeed, we probably have that $X^k(\mathbf{y}, (\text{conv}(\mathbf{C}^i))_{i=1}^n) \subset X^k(\mathbf{y}, (C_i)_{i=1}^n)$. Therefore, other procedures can be used to obtain estimates of $(C_i)_{i=1}^n$. For example, we could use the minimal volume enclosing hyper-spheres of the given data or hyper-ellipsoids that enclose the data. It can be argued, however, that the choice for a particular method for constructing a set that extends the convex hull of the data is rather arbitrary. In the next chapter, we encounter several examples that do allow us to extend the convex hull in an intuitive manner.

7.4. Real-life applications

In this section, we discuss several applications that require the estimation of mixing proportions. In most applications, the sets $C_1, \ldots, C_n$ used to construct $X(\mathbf{y}, (C_i)_{i=1}^n)$ are not given. Instead, these sets are replaced with their estimates $\hat{C}_1, \ldots, \hat{C}_n$. These sets can then be used to compute the estimated feasible set $X(\mathbf{y}, (\hat{C}_i)_{i=1}^n)$, and its projection $X^k(\mathbf{y}, (\hat{C}_i)_{i=1}^n)$.

Notably, close to none (we elaborate on this later) of the publications that we found use the approach that is described in this chapter. Instead, existing applications focus on estimating proportions using a single-point representation of the sources

rather than a set representation. Alternatively, a few authors define priors on the sources and the mixing proportions and subsequently use Bayesian inference procedures to obtain estimates for the mixing proportions.

7.4.1. Detecting fraudulent adulteration of (vegetable) oils

Edible vegetable oils, such as for instance olive oil, are often adulterated with other edible oils for a number of reasons. To some extent, such an adulteration is allowed. However, often there exists a legal limit on the percentage of adulteration. To be able to detect fraudulent levels of adulteration, procedures are needed that provide estimates of the percentage of adulteration in a given oil mixture. As oils are mainly characterized by their fatty acid composition, we can choose $\mathbf{y}$ to be the fatty acid composition vector of such a mixture, and the prototype vectors $\mathbf{c}_1, \ldots, \mathbf{c}_n$ represent the fatty acid composition of the sources (*i.e.* the pure oils) in the LMM. Moreover, a vast amount of literature exists reporting the fatty acid composition of pure (source) oils extracted from plants that were grown under different circumstances. Using this information, $\hat{C}_i$ is defined as the convex hull of all fatty acid vectors reported for the ith oil. Note that we use the convex hull of the observed prototype vectors, rather than the observed prototype vectors themselves to define $\hat{C}_i$. We argue that this is correct, as any element of the convex hull that describes such a source can be obtained by blending some of the observed pure oils. This means that each element in the convex hull represents a pure blend.

We acknowledge that this interesting example was brought to our attention by researchers from the Department of Food Safety and Food Quality of the faculty of Bioscience Engineering of Ghent University. Moreover, within the field of food technology there seems to be an interest in detecting fraudulent levels of adulteration of vegetable oils (see for instance [90, 91, 92, 93, 94]). In most of these publications the focus is mainly on the detection of adulteration using chemical information on the samples as features in a machine learning setting. The research in these papers is mainly data-driven and there is no incorporation of knowledge of the sources or their natural variability. On the other hand, there exists a literature on the decomposition of oil mixtures based on the triacylglyceride (TAG) distribution of both the mixture and the sources (see [95] and references therein). However, the TAG distributions of the sources are generally considered as (noisy observations of) fixed prototype vectors.

7.4.2. Estimating abundance fractions in mixed pixels

Due to the low spatial resolution of hyperspectral sensors used in earth observation studies, the area that is covered by a single pixel often contains several types of land use classes (for instance, forest, agricultural land, urban zone). The

hyperspectral signal that is measured for such a pixel only contains the average amount of radiation emitted through each of the land use classes that is present. As such, a large amount of sub-pixel information is lost. However, we can assume that the proportion of the area covered by the ith land use class is equal to the relative contribution of this class to the observed signal (which is in fact the LMM assumption). Here, $\mathbf{y}$ is the observed hyperspectral signal, and $\mathbf{c}_1, \ldots, \mathbf{c}_n$ represent the hyperspectral signals of the prototypes. As before, $\hat{C}_i$ can be constructed as the convex hull of a set of hyperspectral signals that are known to belong to the ith class. To obtain these prototype signals, several options exist. As the majority of pixels in an image represent pure land cover classes that are easily recognizable on sight, the source signals can be hand picked. As a second option, several algorithmic procedures have been developed that allow for an automated determination of prototype vectors (often called endmembers in this field) in a given image (for instance [96, 97]). However, due to the high dimensionality of the signals, problems may arise due to the presence of noise in the data. We return on this issue in the next chapter.

Interestingly, there exists an extensive literature on what is called spectral unmixing of mixed pixels, see for instance [89, 98, 99], that is closely related to the description given above. In these papers, the abundance fractions of the different land use classes in mixed pixels are estimated using the observed spectral signature of these pixels. Mostly, the spectral signatures of the pure classes (endmembers) are given and the LMM assumption is used to obtain an estimate of these abundance fractions in a least squares approach (we elaborate on this later).

7.4.3. Applications in the earth sciences

In several branches of the earth sciences, such as geochemistry, petrology and mineralogy, there is an interest in decomposing a series of observations (for example sediments) into the proportional contribution of several (assumed) sources of parent material. The underlying assumption here is that observed sediments are mixtures of parent material. Often, the representation of this parent material is assumed unknown. Therefore, unsupervised data mining techniques are used to provide data-driven decomposition of the observations into several endmembers (what we call prototype vectors) and the proportional contribution of these endmembers to each of the samples. An interesting summary of this work can be found in [100].

7.4.4. Applications in (molecular) biology and medicine

In the field of (applied) biology and bioinformatics, several studies can be found that describe the estimation of mixing coefficients. For instance in [101, 102], the

authors use differential gene expression, registered by micro-arrays, to estimate the relative amount of cells in a specific stage of the cell cycle. When studying the chemical composition of mixtures with spectrometric techniques, the LMM is often used to estimate the relative abundance of a chemical component. To be able to provide such an estimate, researchers sometimes use libraries of source spectra (one spectrum for each source) [103].

7.4.5. Applications: Concluding remarks

As illustrated in this section, the decomposition of mixtures has applications in several research areas. These applications share the goal of estimating the proportional contribution of several sources (or endmembers) to a mixture. Some authors start from a single-point description of these sources, whereas others use probabilistic descriptions of these sources or use a completely data-driven methodology to compensate for the absence of prior knowledge on the sources. Unfortunately, depending on the application domain, highly similar problems are often given different names. During our literature search, we came across names such as spectral mixture analysis, spectral deconvolution, deconvolution of mixtures, spectral unmixing, endmember modeling, unmixing, quantification, ... The variety in the naming of the methods that have been proposed is even more exuberant. Moreover, the description of these methodologies is often strongly problem oriented. As a result, the number of cross-references between these application domains is extremely small. On the other hand, within the more general (less problem-specific) data analysis literature, we could not find publications that deal with these problems in a fundamental manner.

Because of the issues raised above, it is hard to verify the novelty of any problem setting or analysis procedure. In the following section, we will describe the most important solution strategies that we found in literature without reference to a specific application.

7.5. Related work

In this section, we will describe the most important solution strategies that we found in literature in a mathematical manner.

7.5.1. The LMM assumption

We start this related work section by pointing at the subtle difference between x_i in the LMM, and the proportional contribution of the ith component to a mixture. When the LMM is used, both are assumed equal. However, when the mixing

process is nonlinear in the representation that is chosen, this assumption is wrong. In applied research domains, such as for instance remote sensing, nonlinear mixing models have been proposed that allow to model specific nonlinear mixing processes, see [89] and the references therein. Moreover, it has been shown that in some cases, inappropriate use of the LMM may cause misleading estimates of the mixing coefficients [89]. In blind source separation[4], nonlinear mixing models have been studied as well [104]. Following the majority of research in mixture analysis, we will be using the LMM in this chapter. This means that most conclusions are only valid when the LMM assumption is correct, or at least provides a good approximation to the real mixing process.

In several settings, the LMM assumption can easily be derived from (and verified by) our knowledge about the system under study. For example, in the introductory example, the linear mixing model is a natural model to use. However, in more complicated settings, the knowledge that is required to verify the LMM assumption is not available. In those cases, the LMM assumption can be verified by means of an experiment. For example, mixtures can be created artificially by mixing multiple sources. As the mixing proportions are known in this case, they can be used as ground truth. Subsequently, to construct a representation of the mixture and the sources, several characteristics are measured (the observed values for these characteristics are the source vectors and the mixture vector). Lastly, estimates for the mixing proportions can be obtained using the LMM and these estimates can be compared to the ground truth. Based on that a comparison, it can be decided if the LMM assumption is valid.

7.5.2. Statistical estimation procedures

Below, we briefly describe two popular statistical approaches for estimating the mixing coefficients in the LMM. First, we consider a constrained least squares estimation approach, which is very popular in several applied domains (for instance, oil mixtures in food industry [95], remote sensing [98] and medicine [105]). Generally, least squares methodologies assume that an observed mixture vector $\mathbf{y}$ can be written as a linear combination of n prototype vectors plus an additional error term, leading to the following model

$$\tilde{\mathbf{y}} = \sum_{i=1}^{n} x_i \, \mathbf{c}_i + \epsilon \,, \tag{7.3}$$

which differs from the LMM (7.1) by a $q \times 1$ error vector. When $\mathbf{y}$ and $\mathbf{c}_1, \ldots, \mathbf{c}_n$ are given, the constrained least squares estimator $\hat{\mathbf{x}}$ is the optimal point of the

[4] Blind source separation is a branch in signal processing/machine learning that focuses on the data-driven decomposition of mixtures of signals into their constituents. Here matrix factorization methods are often used.

following optimization problem.

$$\begin{array}{ll} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} & \|\mathbf{y} - \sum_{i=1}^n x_i\,\mathbf{c}_i\|_2^2 \\ \text{subject to} & \sum_{i=1}^n x_i = 1\,, \\ & x_1,\ldots,x_n \geq 0\,. \end{array}$$

This formulation[5] mainly applies to situations where $q >> n$, such that the solution of the optimization problem is uniquely defined. Three main differences exist between our set estimator described in the introduction and the setting to which the constrained least squares estimator applies. Firstly, the least squares approach requires a single-point description of the sources; as such, our approach can be seen as more generally applicable. Secondly, the least squares approach is generally designed for high-dimensional, (and thirdly) noisy data. On the other hand, our set-based estimator mainly applies to situations where $q < n$. Moreover, up to now, our set estimator is defined in a noise-free setting. From Eq. (7.2) it can be seen that, as the dimensionality q increases, the cardinality of the set $X(\tilde{\mathbf{y}}, (C_i)_{i=1}^n)$ will generally decrease. Moreover, it is not hard to show that as $q \to \infty$ we have that $X(\tilde{\mathbf{y}}, (C_i)_{i=1}^n) \to \emptyset$. This will typically occur in a setting where observed (high-dimensional) spectra are used to represent the mixture. It is clear that, in these situations, the current version of our set-based estimator will be rather useless. Fortunately, we can deal with this issue in a methodological manner. We will return to this issue in the following chapter.

Bayesian modeling approaches have been used to estimate mixing coefficients. Mainly for the analysis and deconvolution of spectral data, Bayesian modeling strategies have proven to be very effective (see for instance [103] for Bayesian modeling with NMR data, or [106] for an application on Raman spectroscopy). In Astle *et al.* [103], a Bayesian procedure is proposed for predicting the abundance of metabolites in a complex biological mixture based on the H-NMR spectrum of that mixture. The authors use the LMM (7.3). To define priors over the sources, a library containing H-NMR profiles of frequently occurring metabolites, is used. Moreover, their method takes into account that some metabolites may not be present in the library. Estimates (or credibility intervals) for the abundance of the metabolites of interest are obtained by likelihood maximization. The work of Astle *et al.* [103] is related to the interval estimates that were proposed in the introduction, as the data driven construction of priors in [103] provides an imprecise, probabilistic description of the sources (the metabolites). In the same spirit, the sets $C_1, \ldots, C_n$ provide a set-based alternative to the imprecise description of sources.

Finally, we mention the link between this chapter and blind source separation

[5] Note that this constrained least squares problem falls within the class of convex, linearly constrained least squares problems and can be solved efficiently.

(BSS). Even though the aim of blind source separation (*i.e.* the decomposition of an (observed) multivariate signal into a set of (independent) source signals), is different from the one we discuss here, the use of the LMM is highly similar. Moreover, BSS is sometimes used to predict mixing coefficients in settings where a description of the sources is missing, but can be derived from data. Examples include non-negative matrix factorization [107] or Bayesian positive source separation [108].

7.5.3. Point estimators, probability estimators or set estimators

The constrained least squares estimator and the Bayesian modeling approach can be related to our set estimator. We argue that these estimators mainly differ in the assumptions that are made. As such, these methods can complement each other, rather than being competitive. Firstly, the restricted least squares estimator implies that the sources are known exactly, and that the observed mixture vector $\tilde{\mathbf{y}}$ is a noisy observation of the true mixture vector $\mathbf{y}$, as expressed by Eq. (7.3). In this perspective, the constrained least squares method provides a point estimate of the mixing coefficients. The uncertainty on the estimate $\hat{\mathbf{x}}$ only results from the noise in the observation. If needed, by making several assumptions about the noise term, this uncertainty can be translated into a confidence interval.

Secondly, the Bayesian modeling approach can be seen as a conceptual generalization of the constrained least squares estimator, as it allows to include uncertainty on the sources (in addition to the noise term). Bayesian modeling strategies express this uncertainty by defining priors on the sources. When, as in the motivating examples or in [103], data are available about these sources, these data can be used to define a data-driven prior. The estimated posterior distribution on $\mathbf{x}$, resulting from this modeling procedure, can be used to assess the uncertainty on the mixing coefficients in a probabilistic manner (if needed by using credibility intervals). However, the adequacy of this assessment strongly depends on the appropriateness of the priors that are used.

Thirdly, our set estimator requires that the uncertainty about the sources is described in an epistemic manner (using sets) rather than a probabilistic manner. The set that is obtained has a clear possibilistic interpretation; it represents a range of values for x_i that can explain our observation (*i.e.* possible values), in contrast to the Bayesian approach that gives an interval of likely values (*i.e.* probable values) for the mixing coefficient. From this reasoning, it follows that a good interval estimate for x_k can only be obtained when the sets $\hat{C}_1, \ldots, \hat{C}_n$ provide a good approximation to $C_1, \ldots, C_n$. This contrasts the Bayesian approach, which requires that the distributions over these sets are specified correctly. Figure 7.3 illustrates the connection between the three estimators.

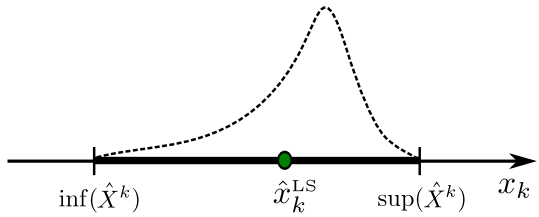


Figure 7.3: Comparison of three estimators: the least squares point estimate $\hat{x}_k^{\mathrm{LS}}$, the Bayesian (estimated) posterior distribution (dotted line), and the set estimate $[\inf(\hat{X}^k), \sup(\hat{X}^k)]$.

7.5.4. An interval estimator in remote sensing

During our search through literature, we came across one publication [109] that deserves special attention here due to its similarity to our set estimator. Even though Bajjouk *et al.* [109] strongly focus on a specific remote sensing application, there are some elements that are strongly related to our approach. More precisely, the authors start from a hyperspectral image, extract a collection of endmembers (prototype vectors using our terminology) and group these endmembers into n classes. Essentially, in this manner they obtain matrices $\mathbf{C}^1 \in \mathbb{R}^{m_1 \times q}, \ldots, \mathbf{C}^n \in \mathbb{R}^{m_n \times q}$ where the rows of the ith matrix represent the endmembers that belong to the ith class (cfr. $\mathbf{C}^1$ and $\mathbf{C}^2$ in Section 7.3.1 of this chapter). Subsequently, for a given mixture vector $\mathbf{y}$, the contribution of the first class to $\mathbf{y}$ is argued to be somewhere in the interval $[a, b]$, where a is the optimal value of a linear program (this linear program is described in detail in Appendix 7.A at the end of this chapter). Similarly, b is the optimal value of the maximization problem with the same objective function and the same (in)equality constraints.

In [110] (different authors), the method introduced in [109] was named *bundle unmixing*. However, in [110], close to no fundamental contributions were made to the method proposed in [109]. These papers have been cited several times but, as far as we know, the methodology was never extended nor applied extensively.

In comparison to our work, we argue that our set estimator is more holistic than the approach of [109]. Firstly, we provide a more profound motivation for our methodology. Secondly, our estimator allows the sources to be described by generic sets, whereas the work in [109] focuses on (specific forms of) polytopes.

7.6. Conclusions and discussion

In this chapter, we proposed the general principle of set-based unmixing of mixture data (Objective III.1). As the examples in this chapter illustrate, the development of unmixing procedures can be of interest to data-analysts and researchers in a

variety of research domains. As a result, multiple methodologies have been proposed that allow to unmix mixture data. However, such methods are generally designed to provide point estimates of the proportional contribution of a series of sources to a mixture and therefore require single-point representations of these sources. Even though these approaches are generally interesting and useful in practice, they have one potential shortcoming. These methods require a description of the sources by means of a point in the Euclidean space or a probability distribution on this space, which is not necessarily available. Therefore, the requirements of our set estimator are less strict. They only require the sources to be described by subsets of the Euclidean space.

As will be proven in the following chapter, the set $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ is a closed interval when the sets $C_1, \ldots, C_n$ are compact and convex. Consequently, $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ is defined by its minimal and its maximal element. This naturally leads to an optimization problem. In the following chapter, we will develop several algorithms that can be used to solve this optimization problem efficiently.

We end this discussion with several open questions regarding the relationship between the unmixing of mixture data and traditional compositional data analysis. It is clear that, at least mathematically, the n-vector $\mathbf{x}$ of proportional contributions of each of n sources to a mixture is a composition. Therefore, it could be argued (based on the reasoning in Chapter 2) that procedures that are used to obtain any 'estimate' of $\mathbf{x}$ should respect the main principles of compositional data analysis. Unfortunately, the relationship between the traditional analysis of compositional data and the methods that have been developed for the unmixing of mixtures remains unclear. For example, it is still unclear how the main principles of compositional data analysis should be related to the (estimates of) the mixing proportions in an LMM context. In this context, we have two seemingly contradictory approaches:

(i) The unmixing problem is often considered as a prediction problem, *i.e.* a predicted composition $\hat{\mathbf{x}} \in \mathbb{S}^n$ should be close to the 'true' vector $\hat{\mathbf{x}} \in \mathbb{S}^n$. To make such a prediction, the mixture vector $\mathbf{y} \in \mathbb{R}^q$ can be used. A traditional machine learning approach would require a dataset of instances $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ to learn a mapping $\mathbf{f} : \mathbb{R}^q \to \mathbb{S}^n$. As the output is a composition, the discussion in Chapter 2 suggests that the loss functions (that operate on compositions) should respect the main principles of compositional data analysis and therefore only use relative information (which translates into the use of ratios).

(ii) In practice, to solve the unmixing problem and to incorporate 'prior knowledge on the sources', the LMM model is often used. Moreover, the LMM is an intuitively appealing model that is very popular in practice. However, we were unable to find an intuitive manner that allows the LMM to be expressed by means of ratios of $\mathbf{x}$. Therefore, it is hard to relate the constrained least

squares estimator of Section 7.5 to the main principles of compositional data analysis.

At this point, we can only conclude that LMM most likely conflicts with the principles of compositional data analysis. Even though both the LMM and compositional data analysis can be justified from a methodological point of view, they remain (at least for us) incompatible in several aspects.

7.A. Computation of $[a, b]$

In this appendix, we elaborate on the computation of the interval $[a, b]$ (representing the set of possible mixing proportions for the kth source) as described in Section 7.5.4. Given the matrices $\mathbf{C}^1 \in \mathbb{R}^{m_1 \times q}, \ldots, \mathbf{C}^n \in \mathbb{R}^{m_n \times q}$ where the rows of the ith matrix represent the endmembers that belong to the ith class, a is the optimal value of the following optimization problem:

$$\begin{aligned} \underset{(\mathbf{z}_i)_{i=1}^n \in \mathbb{R}^{m_i}}{\text{minimize}} \quad & \mathbf{z}_k^\top \mathbf{1}_{m_k} \\ \text{subject to} \quad & \textstyle\sum_{i=1}^n \mathbf{z}_i^\top \mathbf{1}_{m_i} = 1\,, \\ & \textstyle\sum_{i=1}^n {\mathbf{C}^i}^\top \mathbf{z}_i = \mathbf{y}\,. \\ & z_i \geq \mathbf{0}_{m_i}\,. \end{aligned}$$

Notably, this approach can be seen as a special case of our set estimator. Based on the results that we have introduced so far, this may not seem a trivial result. However, using some of the results obtained in the following chapter, it can easily be shown that the method proposed in [109] coincides with the case where $C_1, \ldots, C_n$ are convex polytopes. More precisely, in that case, when $\mathbf{C}^i$ is a matrix of which the rows contain the coordinates of the vertices of C_i, we have that $X^k(\mathbf{y}, (C_i)_{i=1}^n) = [a, b]$. Therefore, the method described in [109] can be used to compute $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ in the case that $C_1, \ldots, C_n$ are convex polytopes of which the corner points are given (or can easily be obtained).

8 Optimization procedures for set-based unmixing

8.1. Introduction

Given a mixture with *mixture vector* $\mathbf{y} \in \mathbb{R}^q$ and a collection of n sources described by sets $C_1, \ldots, C_n \subset \mathbb{R}^q$, we defined (in Chapter 7) the set $X(\mathbf{y}, (C_i)_{i=1}^n)$ and referred to it as a set estimator of the *proportional contribution* of the sources the mixture. Moreover, we defined $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ as the projection of this set on the kth dimension. However, to be useful in practice, X^k should be representable in a compact manner. As X^k is a set of scalars, we could partially represent it by its infimum and its supremum. However, these values do not necessarily provide a complete characterization of X^k. Indeed, in general there might exist values $a \in [\inf(X^k), \sup(X^k)]$ for which it holds that $a \notin X^k$. Nevertheless, as a partial characterization, the infimum and supremum can be interesting. Moreover, $\inf(X^k)$ can easily be seen to be the optimal value of the following optimization problem (without loss of generality, we will assume that $k = 1$):

$$
\begin{array}{llll}
\mathcal{M}_1: & \underset{\mathbf{x}\in\mathbb{R}^n, (\mathbf{c}_i)_{i=1}^n \in \mathbb{R}^q}{\text{minimize}} & x_1 & \\
& \text{subject to} & \mathbf{y} = \sum_{i=1}^n x_i\, \mathbf{c}_i\,, & \\
& & 1 = \sum_{i=1}^n x_i\,, & \\
& & x_i \geq 0\,, & \text{for } i = 1, \ldots, n\,, \\
& & \mathbf{c}_i \in C_i\,, & \text{for } i = 1, \ldots, n\,.
\end{array}
$$

We will refer to this mathematical optimization problem as $\mathcal{M}_1$. Similarly, $\sup(X^k)$ can easily be seen to be the optimal value of the following optimization problem:

$$
\begin{array}{llll}
\mathcal{M}_2: & \underset{\mathbf{x}\in\mathbb{R}^n, (\mathbf{c}_i)_{i=1}^n \in \mathbb{R}^q}{\text{maximize}} & x_1 & \\
& \text{subject to} & \mathbf{y} = \sum_{i=1}^n x_i\, \mathbf{c}_i\,, & \\
& & 1 = \sum_{i=1}^n x_i\,, & \\
& & x_i \geq 0\,, & \text{for } i = 1, \ldots, n\,, \\
& & \mathbf{c}_i \in C_i\,, & \text{for } i = 1, \ldots, n\,.
\end{array}
$$

In this chapter we will show that for compact and convex sets $C_1, \ldots, C_n \subset \mathbb{R}^q$, it holds that $X^k = [\min(X^k), \max(X^k)]$. As a result, solving the optimization

problems above then leads to a complete characterization of X^k. Additionally, we develop an efficient optimization scheme that can be used to find these extreme points. Moreover, as argued in the previous chapter, $\mathbf{y}$ and $(C_i)_{i=1}^n$ will not be directly available in practical situations. Instead, we can only use a noisy observation $\tilde{\mathbf{y}}$ and indirectly derived (possibly noisy) estimates $(\hat{C}_i)_{i=1}^n$ of the true sources. In Sections 8.5 and 8.6 we will address this problem and potential consequences. Finally, we consider several interesting cases where the sources are described by non-convex sets.

Note 1: A complete characterization of X^k requires both $\min(X^k)$ and $\max(X^k)$ to be computed. However, as the computation of $\min(X^k)$ is similar to the computation of $\max(X^k)$, we can often limit our analysis to one of the two.

Note 2: Unless stated differently, we will assume that $C_1, \ldots, C_n$ are compact and convex subsets of $\mathbb{R}^q$.

Note 3: The mixture vector $\mathbf{y}$ and the sets $C_1, \ldots, C_n$ can be seen as data or parameters of the optimization problems above. Naturally, the optimal points of these optimization problems depend on the values of these parameters. When we want to stress that we are using specific values for $\mathbf{y}$ and $C_1, \ldots, C_n$, this will be indicated using the following notation $\mathcal{M}_1(\mathbf{y}, (C_i)_{i=1}^n)$.

Note 4: To ensure that $\mathcal{M}_1(\mathbf{y}, (C_i)_{i=1}^n)$ and $\mathcal{M}_2(\mathbf{y}, (C_i)_{i=1}^n)$ are feasible optimization problems, $\mathbf{y}$ should be an element of the convex hull of $\bigcup_{i=1}^n C_i$. Symbolically, this is denoted as $\mathbf{y} \in \operatorname{conv}\left(\bigcup_{i=1}^n C_i\right)$.

Note 5: When $\mathbf{y} \in C_1$, it is trivial to see that the optimum of $\mathcal{M}_2$ is equal to 1. Therefore, we will often exclude this case from our analysis.

8.2. A brute force search for $\min(X^k)$ and $\max(X^k)$

Due to the presence of bilinear equality constraints, the optimization problems presented before are not convex. This means that, in general, directly solving $\mathcal{M}_1$ and $\mathcal{M}_2$ is not computationally tractable. Even finding a feasible point is potentially very hard. Disregarding these potential threats, it can be attempted to (locally) solve these optimization problems. Interestingly, when $C_1, \ldots, C_n$ are convex polytopes, $\mathcal{M}_1$ and $\mathcal{M}_2$ are instances of the class of generalized bilinear programs. To see this, recall from Chapter 2 that a convex polytope can be represented by a set of inequalities. Therefore, all constraints in $\mathcal{M}_1$ and $\mathcal{M}_2$ are linear or bilinear, ensuring that these programs are special cases of the generalized bilinear program [23].

The generalized bilinear program (GBP) is non-convex and can have multiple local optima, making it a hard problem to solve globally. Most research on global optimization of the generalized bilinear programming problem has focused on

branch-and-bound like approaches [23, 111]. Even though these methods mostly guarantee to find the global optimum within a finite number of iterations, the computing time needed to obtain the optimum still increases exponentially with the problem size. As such, globally solving the generalized bilinear program in its most general form using branch-and-bound approaches is only tractable when both q and n are small. To verify whether the aforementioned time-complexity issues are of any practical concern, a branch-and-bound strategy was implemented in Matlab. From this exercise, we concluded that these theoretical time-complexity issues are of practical concern as well.

Fortunately, as we will show next, $\mathcal{M}_1$ and $\mathcal{M}_2$ are specific cases of the generalized bilinear program. More precisely, $\mathcal{M}_1$ and $\mathcal{M}_2$ have additional structure that can be exploited to derive an efficient optimization procedure.

8.3. A characterization of X^k

As a starting point, we define the following optimization problem:

$$\begin{aligned} \mathcal{M}_3 : \quad & \underset{\alpha,\bar{\alpha}\in\mathbb{R},\mathfrak{c},\bar{\mathfrak{c}}\in\mathbb{R}^q}{\text{maximize}} && \alpha \\ & \text{subject to} && \mathbf{y} = \alpha\,\mathfrak{c} + \bar{\alpha}\,\bar{\mathfrak{c}}\,, \\ & && 1 = \alpha + \bar{\alpha}\,, \\ & && 0 \leq \alpha, \bar{\alpha}\,, \\ & && \mathfrak{c} \in C_1\,, \\ & && \bar{\mathfrak{c}} \in \operatorname{conv}\left(\textstyle\bigcup_{i=2}^n C_i\right)\,. \end{aligned}$$

Proposition 8.1. *Let* $\mathbf{y} \in \operatorname{conv}(\bigcup_{i=1}^n C_i)$*, let* x_1^* *be the optimal value of* $\mathcal{M}_2(\mathbf{y}, (C_i)_{i=1}^n)$ *and* α^* *the optimal value of* $\mathcal{M}_3(\mathbf{y}, (C_i)_{i=1}^n)$*, then we always have that* $x_1^* = \alpha^*$*.*

Proof. Firstly, given a point $(\mathbf{x}, \mathbf{c}_1, \ldots, \mathbf{c}_n)$ that is feasible for $\mathcal{M}_2$, we can construct a feasible point $(\alpha, \bar{\alpha}, \mathfrak{c}, \bar{\mathfrak{c}})$ for $\mathcal{M}_3$ as follows:

$$\alpha = x_1\,, \tag{8.1}$$

$$\bar{\alpha} = 1 - x_1\,, \tag{8.2}$$

$$\mathfrak{c} = \mathbf{c}_1\,, \tag{8.3}$$

$$\bar{\mathfrak{c}} = \sum_{i=2}^n \frac{x_i\,\mathbf{c}_i}{1 - x_1}\,. \tag{8.4}$$

Naturally, for these points, the objective function value of $\mathcal{M}_2$ is equal to the objective function value for $\mathcal{M}_3$.

Given a point $(\alpha, \bar{\alpha}, \mathfrak{c}, \bar{\mathfrak{c}})$ that is feasible for $\mathcal{M}_3$, there exists at least one point $(\mathbf{x}, \mathbf{c}_1, \ldots, \mathbf{c}_n)$ that is feasible for $\mathcal{M}_2$. To see this, we can take

$$x_1 = \alpha\,, \tag{8.5}$$

$$\mathbf{c}_1 = \mathfrak{c}\,. \tag{8.6}$$

As $\bar{\mathfrak{c}} \in \operatorname{conv}(\bigcup_{i=2}^n C_i)$, there exists at least one tuple $(x_2', \ldots, x_n') \in \mathbb{S}^{n-1}$ and $\mathbf{c}_2 \in C_2, \ldots, \mathbf{c}_n \in C_n$ such that $\bar{\mathfrak{c}} = \sum_{i=2}^n x_i' \mathbf{c}_i$. Consequently, we can choose $(x_2, \ldots, x_n) = (1-\alpha)(x_2', \ldots, x_n')$. Here as well, we have that for these points, the objective function value of $\mathcal{M}_2$ is equal to the objective function value for $\mathcal{M}_3$.

This means that, for each feasible point for $\mathcal{M}_2$ (resp. $\mathcal{M}_3$), there exists at least one feasible point for $\mathcal{M}_3$ (resp. $\mathcal{M}_2$) such that the objective function values are equal. □

The former proposition implies that the solution of $\mathcal{M}_3$ leads to a solution of $\mathcal{M}_2$. As a result, we can focus on solving $\mathcal{M}_3$. However, $\mathcal{M}_3$ can easily be seen as an instance of $\mathcal{M}_2$ with only two sources. The first source is represented by C_1 and the second source is represented by $\bar{C} = \operatorname{conv}(\bigcup_{i=2}^n C_i)$. This means that $\mathcal{M}_2(\mathbf{y}, (C_i)_{i=1}^n)$ and $\mathcal{M}_2(\mathbf{y}, (C_1, \bar{C}))$ are equivalent. Therefore, we can simply focus on solving $\mathcal{M}_2(\mathbf{y}, (C_1, C_2))$ (where $\bar{C}$ is replaced by C_2 merely for notational purposes). For notational convenience, we formally define the mathematical optimization problem $\mathcal{M}_4$ that is a specific case of $\mathcal{M}_2$ with only two sources.

$$\begin{aligned}
\mathcal{M}_4 : \quad & \underset{x_1 \in \mathbb{R}, x_2 \in \mathbb{R}, \mathbf{c}_1 \in \mathbb{R}^q, \mathbf{c}_2 \in \mathbb{R}^q}{\text{maximize}} && x_1 \\
& \text{subject to} && \mathbf{y} = x_1\,\mathbf{c}_1 + x_2\,\mathbf{c}_2\,, \\
& && 1 = x_1 + x_2\,, \\
& && 0 \leq x_1, x_2\,, \\
& && \mathbf{c}_1 \in C_1\,, \\
& && \mathbf{c}_2 \in C_2\,.
\end{aligned}$$

In what follows, we derive an optimization scheme for $\mathcal{M}_4$. To formulate this scheme, and show its convergence to the global optimum, the following definitions will be used.

Definition 8.1. *A scalar x_1 is called $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$-feasible (or shorthand $\mathcal{M}_4$-feasible) if there exists a point $(x_1, x_2, \mathbf{c}_1, \mathbf{c}_2) \in \mathbb{R} \times \mathbb{R} \times \mathbb{R}^q \times \mathbb{R}^q$ that is a feasible point of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$.*

Definition 8.2. *A couple $(\mathbf{c}_1, \mathbf{c}_2) \in \mathbb{R}^q \times \mathbb{R}^q$ is called $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$-feasible (or shorthand $\mathcal{M}_4$-feasible) if there exists a point $(x_1, x_2, \mathbf{c}_1, \mathbf{c}_2) \in \mathbb{R} \times \mathbb{R} \times \mathbb{R}^q \times \mathbb{R}^q$ that is a feasible point of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$.*

These definitions are used in the following propositions.

Proposition 8.2. *For every $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$-feasible couple $(\mathbf{c}_1, \mathbf{c}_2) \in \mathbb{R}^q \times \mathbb{R}^q$, there exists exactly one couple $(x_1, x_2) \in [0,1]^2$ such that $(x_1, x_2, \mathbf{c}_1, \mathbf{c}_2)$ is a feasible point of $\mathcal{M}_4$. Moreover, for this couple we have*

$$x_1 = \frac{\|\mathbf{c}_2 - \mathbf{y}\|_2}{\|\mathbf{c}_1 - \mathbf{y}\|_2 + \|\mathbf{c}_2 - \mathbf{y}\|_2}\,.$$

Proof. This proof is a trivial consequence of the LMM. □

Proposition 8.3. *If both scalars $\bar{x}_1$ and $\dot{x}_1$ are $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$-feasible, then any scalar $z = \alpha\,\bar{x}_1 + (1-\alpha)\,\dot{x}_1$, with $\alpha \in [0,1]$, is $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$-feasible as well.*

Proof. In this proof, we will assume that $\mathbf{y} = \mathbf{0}_q$. This does not affect the generality of the proof as this case can always be obtained by translating the coordinate system (which does not affect the optimal value).

As $\bar{x}_1$ and $\dot{x}_1$ are $\mathcal{M}_4$-feasible, there exist at least two points $(\bar{x}_1, \bar{x}_2, \bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$ and $(\dot{x}_1, \dot{x}_2, \dot{\mathbf{c}}_1, \dot{\mathbf{c}}_2)$ that are feasible points of $\mathcal{M}_4$. Let us define the vector functions $\mathbf{h}_1 : [0,1] \to \mathbb{R}^q$ and $\mathbf{h}_2 : [0,1] \to \mathbb{R}^q$ as follows

$$\mathbf{h}_1(\beta) = \bar{\mathbf{c}}_1\,\beta + \dot{\mathbf{c}}_1\,(1-\beta)\,,$$

$$\mathbf{h}_2(\beta) = \bar{\mathbf{c}}_2\,\beta + \dot{\mathbf{c}}_2\,(1-\beta)\,.$$

For any $\beta \in [0,1]$, we have that $\mathbf{h}_1(\beta) \in C_1$ and $\mathbf{h}_2(\beta) \in C_2$. Moreover, as $(\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$ (resp. $(\dot{\mathbf{c}}_1, \dot{\mathbf{c}}_2)$) is an $\mathcal{M}_4$-feasible couple, we have that

$$\bar{\mathbf{c}}_2 = \bar{\mathbf{c}}_1\,\bar{t} \qquad \text{and} \qquad \dot{\mathbf{c}}_2 = \dot{\mathbf{c}}_1\,\dot{t}\,, \tag{8.7}$$

for some scalars $\bar{t}, \dot{t} < 0$. Additionally, these equalities can be used to rewrite $\mathbf{h}_2$ as follows

$$\mathbf{h}_2(\beta) = \bar{\mathbf{c}}_1\,\bar{t}\,\beta + \dot{\mathbf{c}}_1\,\dot{t}\,(1-\beta)\,.$$

Now consider the vector

$$\mathbf{m} = \left(x_1(\beta), x_2(\beta), \mathbf{h}_1(\beta), \mathbf{h}_2\left(\dot{t}\left(\frac{1-\beta}{\beta}\bar{t} + \dot{t} \right)^{-1} \right) \right)^\top,$$

where

$$x_1(\beta) = \frac{1}{1 - \frac{(1-\beta)\,\bar{t} + \beta\,\dot{t}}{\bar{t}\,\dot{t}}}\,, \qquad \text{and} \qquad x_2(\beta) = 1 - x_1(\beta)\,.$$

It can easily be verified for any $\beta \in [0,1]$ that $\mathbf{m}$ is a feasible point of $\mathcal{M}_4$.

Moreover, by applying Proposition 8.2, we obtain that

$$x_1(0) = \bar{x}_1 = \frac{1}{1 - 1/\bar{t}}\,, \qquad \text{and} \qquad x_1(1) = \dot{x}_1 = \frac{1}{1 - 1/\dot{t}}\,. \tag{8.8}$$

As $x_1(\beta)$ is a continuous, monotone function of β, we know that (from the intermediate value theorem) the range of $x_1(\beta)$ is $[\bar{x}, \dot{x}]$. This means that, for any $z = \alpha\,\bar{x}_1 + (1-\alpha)\,\dot{x}_1$, there exists a $\beta \in [0,1]$ such that $z = x_1(\beta)$. □

The above proposition implies that for any convex sets C_1 and C_2, the set X^k is an interval. Moreover, we can efficiently verify whether a scalar $x \in [0,1]$ is $\mathcal{M}_4$-feasible using the following convex feasibility problem:

$$\begin{aligned} \underset{t\in\mathbb{R},\mathbf{c}_1\in\mathbb{R}^q,\mathbf{c}_2\in\mathbb{R}^q}{\text{minimize}} \quad & t \\ \text{subject to} \quad & -t\,\mathbf{1}_q \leq x\,\mathbf{c}_1 + (1-x)\,\mathbf{c}_2 - \mathbf{y} \leq t\,\mathbf{1}_q\,, \\ & \mathbf{c}_1 \in C_1\,, \\ & \mathbf{c}_2 \in C_2\,. \end{aligned}$$

Let t^* be the solution of this feasibility problem; we have that x is $\mathcal{M}_4$-feasible if and only if $t^* = 0$. Moreover, as X^k is an interval, we can use a bisection algorithm to find $\inf(X^k)$ and $\sup(X^k)$. For example, the following procedure can be used to find $\sup(X^k)$.

1. Select an $\mathcal{M}_4$-feasible scalar x and let $a := x$ and $b := 1$. Lastly, choose a tolerance parameter ϵ.
2. **If** $|a - b| \leq \epsilon$, quit and return a, **else** compute $c := (a+b)/2$.
3. Check whether c is an $\mathcal{M}_4$-feasible scalar.
   - **If** c is an $\mathcal{M}_4$-feasible scalar, set $a := c$ and go to step 2.
   - **Else** set $b := c$ and go to step 2.

To be able to guarantee that this will lead to an efficient procedure, an initial value x that is $\mathcal{M}_4$-feasible is needed. Unfortunately, such a value is generally not given in practice. Moreover, finding such a value seems a non-trivial task. We would like to note that the objective function value of the feasibility problem could potentially be used to find such an initial value. However, we will not consider such an approach in depth.

Conclusion

In this section, the following results were obtained:

- The optimization problem $\mathcal{M}_4$ was introduced and shown to be equivalent to $\mathcal{M}_2$.

- We have shown that, for convex subsets $C_1, \ldots, C_n$ of $\mathbb{R}^q$, $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ is an interval.
- Once an $\mathcal{M}_4$-feasible scalar is given, the interval $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ can be computed efficiently using a bisection algorithm.

8.4. Efficiently computing X^k

In the previous section, we have proven that X^k is an interval. Moreover, this property implies that a bisection approach can be used to find $\inf(X^k)$ and $\sup(X^k)$. In this section, we derive a more in-depth characterization of the solution of $\mathcal{M}_4$ that will allow us to gain further insight into the problem and derive a more general optimization procedure. It should be stated that the main results are rather intuitive and can easily be understood using the basic geometry of the problem. Therefore, the general intuition behind the solution strategy is presented in the following subsection and the technical details are presented in a separate subsection.

8.4.1. An intuitive approach

As a starting point, we consider the case where $q = 1$, which implies that $C_1 = [b_1^l, b_1^u]$ and $C_2 = [b_2^l, b_2^u]$ are intervals. An example of this case is presented in Figure 8.1. We now focus on the computation of $\sup(X^k)$. The geometry behind the computation of $\sup(X^k)$ is rather simple. Recall from Proposition 8.2 that, for a given $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$-feasible couple $(\mathbf{c}_1, \mathbf{c}_2)$, the proportional contribution x_1 of $\mathbf{c}_1$ is

$$x_1 = \frac{\|\mathbf{c}_2 - \mathbf{y}\|_2}{\|\mathbf{c}_1 - \mathbf{y}\|_2 + \|\mathbf{c}_2 - \mathbf{y}\|_2}.$$

Geometrically, the couple $(\mathbf{c}_1, \mathbf{c}_2)$ for which x_1 is maximal, is the couple for which $\mathbf{c}_2$ is as far away from $\mathbf{y}$ as possible and $\mathbf{c}_1$ is as close to $\mathbf{y}$ as possible. In the example in Figure 8.1, this amounts to setting $\mathbf{c}_1 = b_1^u$ and $\mathbf{c}_2 = b_2^u$; as a result, we obtain that $\sup(X^1) = \frac{b_2^u - \mathbf{y}}{b_2^u - b_1^u}$. This closed-form solution is applicable to any case in which $b_1^u < \mathbf{y} \leq b_2^u$ (this is formally shown (and generalized) in the following section). Later in this section, we will see that this simple case serves as a building block for computing $\sup(X^k)$ in settings where the dimensionality q is greater than one.

8.4.2. Several properties of $\mathcal{M}_4$ for $q = 1$

Let $C_1 = [b_1^l, b_1^u]$, $C_2 = [b_2^l, b_2^u]$ and $\mathbf{y} \in \operatorname{conv}(C_1 \cup C_2) \setminus C_1$ (see Notes 4 and 5 in Section 8.1). Let $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ be an optimal point of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$. In this

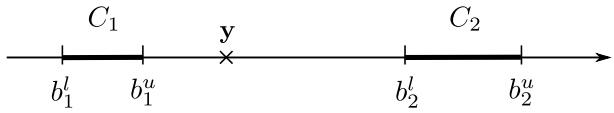


Figure 8.1: Visualization of the search space when $q = 2$. We use the notation $b_1^l = \inf(C_1)$ (resp. $b_2^l = \inf(C_2)$) and $b_1^u = \sup(C_1)$ (resp. $b_2^u = \sup(C_2)$).

section we show that

$$(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*) = \begin{cases} (\frac{\mathbf{y}-b_2^u}{b_1^u-b_2^u}, 1-\frac{\mathbf{y}-b_2^u}{b_1^u-b_2^u}, b_1^u, b_2^u)\,, & \text{if } b_1^u < \mathbf{y} < b_2^u\,, \quad (a) \\ (0, 1, ., \mathbf{y})\,, & \text{if } b_1^u < \mathbf{y} = b_2^u\,, \quad (b) \\ (\frac{\mathbf{y}-b_2^l}{b_1^l-b_2^l}, 1-\frac{\mathbf{y}-b_2^l}{b_1^l-b_2^l}, b_1^l, b_2^l)\,, & \text{if } b_2^l < \mathbf{y} < b_1^l\,, \quad (c) \\ (0, 1, ., \mathbf{y})\,, & \text{if } b_2^l = \mathbf{y} < b_1^l\,, \quad (d) \end{cases}$$

where the dot indicates that there are multiple optimal points. Additionally, we show that there are no (other) locally optimal points.

For cases (b) and (d) it is clear that the points listed are the only feasible points. Therefore, it is trivial that they are optimal.

Lemma 8.4. *Let $C_1 = [b_1^l, b_1^u]$, $C_2 = [b_2^l, b_2^u]$, and $b_1^u < \mathbf{y} < b_2^u$, then every feasible point of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ satisfies LICQ*[1].

Proof. We start by rephrasing $\mathcal{M}_4$ slightly, by eliminating x_2 and transforming it into a more simple form (*i.e.* writing it in the formulation of Definition 3.1). This leads to the following optimization problem:

$$\underset{x_1\in\mathbb{R}, \mathbf{c_1}\in\mathbb{R}^q, \mathbf{c_2}\in\mathbb{R}^q}{\text{minimize}} \; -x_1 \tag{8.9}$$

subject to

$$\begin{aligned} x_1\,\mathbf{c}_1 + (1-x_1)\,\mathbf{c}_2 - \mathbf{y} &= 0\,, && (8.10)\\ -x_1 &\leq 0\,, && (8.11)\\ -1 + x_1 &\leq 0\,, && (8.12)\\ -\mathbf{c}_1 + b_1^l &\leq 0\,, && (8.13)\\ -\mathbf{c}_2 + b_2^l &\leq 0\,, && (8.14)\\ -b_1^u + \mathbf{c}_1 &\leq 0\,, && (8.15)\\ -b_2^u + \mathbf{c}_2 &\leq 0\,. && (8.16) \end{aligned}$$

The partial derivatives of these constraints are listed in the Table 8.1. Additionally,

[1] Linear independence constraint qualification, see Chapter 3 for a definition.

Table 8.1: The second column of this table contains the partial derivatives of constraints (8.10)–(8.16). Additionally, the bullets in the third column indicate which combinations of active constraints are possible. Each column represents a combination of constraints that can be active at the same time (evidently, subsets of these constraints are possible as well).

Constraint number	Gradient vector	Combinations act. constr. $b_1^u < \mathbf{y} < b_2^l$				$b_2^l < \mathbf{y} < b_2^u$	
(8.10)	$(\ \mathbf{c}_1 - \mathbf{c}_2,\ x_1,\ 1 - x_1\ )$	•	•	•	•	•	•
(8.11)	$(\ -1,\ 0,\ 0\ )$					•	•
(8.12)	$(\ 1,\ 0,\ 0\ )$						
(8.13)	$(\ 0,\ -1,\ 0\ )$	•	•				
(8.14)	$(\ 0,\ 1,\ 0\ )$			•	•		
(8.15)	$(\ 0,\ 0,\ -1\ )$	•		•		•	
(8.16)	$(\ 0,\ 0,\ 1\ )$		•		•		•

the bullets in the third column indicate which combinations of active constraints are possible. Each column represents a combination of constraints that can be active at the same time (evidently, subsets of these constraints are possible as well). As can be seen from this table, every possible combination leads to a set of linearly independent gradient vectors. This means that LICQ holds at every feasible point.

The situation $\mathbf{y} = b_2^l$ is not covered in this Table 8.1. However, in that case constraint (8.14) can be eliminated from the optimization procedure. In that case, an analysis that is highly similar to the one presented above shows that the LICQ holds at every feasible point. □

Lemma 8.5. *Let $C_1 = [b_1^l, b_1^u]$, $C_2 = [b_2^l, b_2^u]$, and $b_2^l < \mathbf{y} < b_1^l$, then every feasible point of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ satisfies LICQ.*

Proof. The proof is analogous to the proof of Lemma 8.4. □

Lemma 8.6. *Let $C_1 = [b_1^l, b_1^u]$ and $C_2 = [b_2^l, b_2^u]$, and $b_1^u < \mathbf{y} < b_2^u$. The point $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*) = (\frac{\mathbf{y}-b_2^u}{b_1^u-b_2^u}, 1 - \frac{\mathbf{y}-b_2^u}{b_1^u-b_2^u}, b_1^u, b_2^u)$ is the only KKT point of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$.*

Proof. As a starting point, it is trivial to verify that $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is a feasible point. To simplify the proof, we eliminate x_2 by replacing it with $1 - x_1$ (which follows from the summation constraint), such that $(x_1, x_2, \mathbf{c}_1, \mathbf{c}_2) \equiv (x_1, \mathbf{c}_1, \mathbf{c}_2)$. Moreover, we rewrite the maximization problem in the from of Eqs. (8.9)–(8.16). For this problem, the Lagrangian (L) becomes

$$\begin{aligned} L((x_1, \mathbf{c}_1, \mathbf{c}_2), \boldsymbol{\lambda}, \boldsymbol{\gamma}) \;=\; & -x_1 + \lambda_1(x_1\,\mathbf{c}_1 + \mathbf{c}_2 - x_1\,\mathbf{c}_2 - \mathbf{y}) - \gamma_1(x_1) - \gamma_2(1 - x_1) \\ & -\gamma_3(\mathbf{c}_1 - b_1^l) - \gamma_4(\mathbf{c}_2 - b_2^l) - \gamma_5(b_1^u - \mathbf{c}_1) - \gamma_6(b_2^u - \mathbf{c}_2)\,. \end{aligned}$$

The partial derivatives of the Lagrangian with respect to $\mathbf{c}_1, \mathbf{c}_2$ and x_1 are:

$$\frac{\partial L}{\partial x_1} = -1 + \lambda_1(\mathbf{c}_1 - \mathbf{c}_2) - \gamma_1 + \gamma_2 \,, \tag{8.17}$$

$$\frac{\partial L}{\partial \mathbf{c}_1} = \lambda_1 x_1 - \gamma_3 + \gamma_5 \,, \tag{8.18}$$

$$\frac{\partial L}{\partial \mathbf{c}_2} = \lambda_1(1 - x_1) - \gamma_4 + \gamma_6 \,. \tag{8.19}$$

It can easily be seen that $(x_1^*, \mathbf{c}_k^*, \mathbf{c}_2^*)$ is a feasible point, moreover, at this point, only the constraints $x_1\mathbf{c}_1 + \mathbf{c}_2 - x_1\,\mathbf{c}_2 = \mathbf{y}$, $b_u - \mathbf{c}_1 \geq 0$ and $\bar{b}_u - \mathbf{c}_2 \geq 0$ are active, which means (using the complementarity conditions of the KKT conditions) that $\gamma_1 = \gamma_2 = \gamma_3 = \gamma_4 = 0$. Setting the partial derivatives of the Lagrangian equal to zero, taking into account the complementarity conditions and plugging in $(x_1^*, \mathbf{c}_k^*, \mathbf{c}_2^*)$ leads to the following Lagrange multipliers:

$$\lambda_1 = \frac{1}{\mathbf{c}_1 - \mathbf{c}_2} < 0 \,, \quad \gamma_5 = -\lambda_1 \frac{\mathbf{y} - b_2^u}{b_1^u - b_2^u} > 0 \,, \quad \gamma_6 = -\lambda_1 \frac{\mathbf{y} - b_2^u}{b_1^u - b_2^u} > 0 \,.$$

As such, the KKT conditions hold at $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$, which means that the necessary conditions hold.

We now show that there exist no other points that satisfy the KKT conditions. Firstly, consider a generic point $(x_1^0, \mathbf{c}_1^0, \mathbf{c}_2^0)$. When this point is limited to be in the interior (not at the bounds) of the search space, none of the Lagrange multipliers are zero. It can easily be shown that solving the KKT system in this case leads to Lagrange multipliers, for the inequality constraints, of which at least one is negative. The same holds when only one of the variables is located at its bounds, as well as when both $\mathbf{c}_1^0 = b_1^\ell$ and $\mathbf{c}_2^0 = b_2^\ell$. □

Lemma 8.7. *Let $C_1 = [b_1^l, b_1^u]$, $C_2 = [b_2^l, b_2^u]$ and $b_2^l < \mathbf{y} < b_1^l$, then the point $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*) = (\frac{\mathbf{y}-b_2^l}{b_1^l - b_2^l}, 1 - \frac{\mathbf{y}-b_2^l}{b_1^l - b_2^l}, b_1^l, b_2^l)$ is the only KKT point of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$.*

Proof. The proof is analogous to the proof of Lemma 8.6. □

Proposition 8.8. *Let $C_1 = [b_1^l, b_1^u]$, $C_2 = [b_2^l, b_2^u]$ and $b_1^u < \mathbf{y} < b_2^u$, then the point $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*) = (\frac{\mathbf{y}-b_2^u}{b_1^u - b_2^u}, 1 - \frac{\mathbf{y}-b_2^u}{b_1^u - b_2^u}, b_1^u, b_2^u)$ is globally optimal for $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$. Moreover, there are no other locally optimal points.*

Proof. From Lemma 8.4, we have that the linear independence constraint qualification (LICQ) holds at every feasible point. As such, any local minimizer of $\mathcal{M}_4$ must necessarily satisfy the KKT conditions. Moreover, from Lemma 8.6, we have that $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is the only KKT point. Therefore, it is the only candidate for being an optimal point.

We now use the reformulation of $\mathcal{M}_4$ in Lemma 8.4. We have that only constraints (8.10), (8.14) and (8.16) are active at $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$. Using the Table 8.1, it is easy to

show that the tangent cone to the feasible set (Definition 3.18) is given by

$$\mathbb{F}_1(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*) = \mathbf{0}_3 \,.$$

Therefore, the second order sufficient conditions (Proposition 3.3) for a locally optimal point trivially hold. As there are no other KKT points, $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is the globally optimal point of $\mathcal{M}_4$. □

Proposition 8.9. *Let $C_1 = [b_1^l, b_1^u]$, $C_2 = [b_2^l, b_2^u]$ and $b_2^l < \mathbf{y} < b_1^l$, then the point $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*) = (\frac{\mathbf{y}-b_2^l}{b_1^l-b_2^l}, 1 - \frac{\mathbf{y}-b_2^l}{b_1^l-b_2^l}, b_1^l, b_2^l)$ is globally optimal for $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$. Moreover, there are no other locally optimal points.*

Proof. The proof is analogous to the proof of Proposition 8.8. □

8.4.3. Several properties of $\mathcal{M}_4$ for $q > 1$

The results that were obtained above can now be used to characterize the solution of $\mathcal{M}_4$ when $q > 1$. To gain insight into this generalized case, we will often resort to the geometrical interpretation of the problem for $q = 2$. Nevertheless, the results obtained here are not limited to $q = 2$. As a running example, we will use the case presented in Figure 8.2(a). Recall the discussion in Section 8.4.1 where it was stated that, for a given $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$-feasible couple $(\mathbf{c}_1, \mathbf{c}_2)$, the proportional contribution x_1 of $\mathbf{c}_1$ is

$$x_1 = \frac{\|\mathbf{c}_2 - \mathbf{y}\|_2}{\|\mathbf{c}_1 - \mathbf{y}\|_2 + \|\mathbf{c}_2 - \mathbf{y}\|_2} \,.$$

Geometrically, the couple $(\mathbf{c}_1, \mathbf{c}_2)$ for which x_1 is maximal, is the couple for which $\mathbf{c}_2$ is as far away from $\mathbf{y}$ as possible and $\mathbf{c}_1$ is as close to $\mathbf{y}$ as possible. Naturally, for $(\mathbf{c}_1, \mathbf{c}_2)$ to be feasible in the case where $q > 1$, $\mathbf{c}_1$ and $\mathbf{c}_2$ should be collinear with $\mathbf{y}$. This complicates the analysis of the problem. Nevertheless, this geometrical interpretation leads to insights into the problem.

We start our analysis by formally introducing the following notation for a line segment and a line; for any two distinct vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^q$, we will denote

$$\begin{aligned} \overline{(\mathbf{a}, \mathbf{b})} &= \{\mathbf{v} \in \mathbb{R}^q \mid (\exists\, t \in [0,1])\,(\mathbf{v} = t\,\mathbf{a} + (1-t)\,\mathbf{b})\}\,, \\ \underline{(\mathbf{a}, \mathbf{b})} &= \{\mathbf{v} \in \mathbb{R}^q \mid (\exists\, t \in \mathbb{R})\,(\mathbf{v} = \mathbf{a} + t\,(\mathbf{a} - \mathbf{b}))\}\,. \end{aligned}$$

Note that, when fixing $\mathbf{c}_1 \in C_1$ and $\mathbf{c}_2 \in C_2$ such that $\mathbf{y} \in \overline{(\mathbf{c}_1, \mathbf{c}_2)}$, the solution of $\mathcal{M}_4$ directly follows from Proposition 8.8. Moreover, we define several sets that will be used in the reasoning hereafter:

$$C_1' = \{\mathbf{c}_1 \in C_1 \mid (\exists\, \mathbf{c}_2 \in C_2)\,(\exists t \in [0,1])(\mathbf{y} = t\,\mathbf{c}_1 + (1-t)\,\mathbf{c}_2)\}$$

$$C_2' = \{\mathbf{c}_2 \in C_2 \mid (\exists\, \mathbf{c}_1 \in C_1)\,(\exists t \in [0,1])(\mathbf{y} = t\,\mathbf{c}_1 + (1-t)\,\mathbf{c}_2)\}$$

For an illustration, see Figure 8.2(b). Note that C_1' is a subset of C_1 that is obtained by removing all elements from C_1 that can never occur in feasible points (due to the bilinear equality constraint). Consequently, replacing C_1 by C_1' in the optimization problem will not reduce the feasible space. Next, we define the convex sets $\bar{C}_1$ and $\bar{C}_2$:

$$\begin{aligned}\bar{C}_1 &= \{\mathbf{c} \in \mathbb{R}^q \mid (\exists\, \mathbf{c}_1 \in C_1')\,(\exists t \in [1,\infty[)\,(\mathbf{c} - \mathbf{y} = t\,(\mathbf{c}_1 - \mathbf{y}))\}\,,\\ \bar{C}_2 &= \{\mathbf{c} \in \mathbb{R}^q \mid (\exists\, \mathbf{c}_2 \in C_2')\,(\exists t \in [1,\infty[)\,(\mathbf{c} - \mathbf{y} = t\,(\mathbf{c}_2 - \mathbf{y}))\}\,.\end{aligned}$$

For an illustration, see Figure 8.2(c). Finally, we define the set C_1^*:

$$C_1^* = \{\mathbf{c} \in C_1' \mid \neg\,((\exists\, \mathbf{c}' \in C_1')\,(\exists t \in]0,1])\,(\mathbf{c}' = t\,\mathbf{y} + (1-t)\,\mathbf{c}))\}\,.$$

From this definition, it is clear that C_1^* is a subset of the boundary of C_1'. For an illustration see Figure 8.2(d).

Let us now fix $\dot{\mathbf{c}}_2 \in C_2'$, the couple $(\mathbf{c}_1, \dot{\mathbf{c}}_2)$ is an $\mathcal{M}_4$-feasible couple if and only if $\mathbf{c}_1 \in \underline{(\mathbf{y}, \dot{\mathbf{c}}_2)} \cap C_1'$, which is a line segment. Moreover, from Proposition 8.8, it follows that on this line segment the point $\mathbf{c}_1$ that maximizes the objective function can be found at the endpoint that is closest to $\mathbf{y}$. As such, if $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is locally optimal, then $\mathbf{c}_1^*$ will be located at the boundary of C_1. More precisely, we have that $\mathbf{c}_1^* \in C_1^*$.

Combining the reasoning above with Proposition 8.2 allows the following reformulation of the original optimization problem:

$$\begin{aligned}\mathcal{M}_5 : \quad & \underset{x_1 \in \mathbb{R}, \mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^q}{\text{maximize}} && \frac{\|\mathbf{c}_2 - \mathbf{y}\|_2}{\|\mathbf{c}_1 - \mathbf{y}\|_2 + \|\mathbf{c}_2 - \mathbf{y}\|_2}\\ & \text{subject to} && \mathbf{y} = x_1\,\mathbf{c}_1 + (1-x_1)\,\mathbf{c}_2\,,\\ & && 0 \le x_1\,,\\ & && x_1 \le 1\,,\\ & && \mathbf{c}_1 \in C_1^*\,,\\ & && \mathbf{c}_2 \in C_2'\,.\end{aligned}$$

This optimization problem can be still be simplified. More precisely, for a fixed vector $\mathbf{c}_2 \in \bar{C}_2$, there exists only one $\mathbf{c}_1 \in C_1^*$ such that the bilinear equality constraint of $\mathcal{M}_5$ holds. As such, $\mathbf{c}_1$ can be written as a (vector) function of $\mathbf{c}_2$.

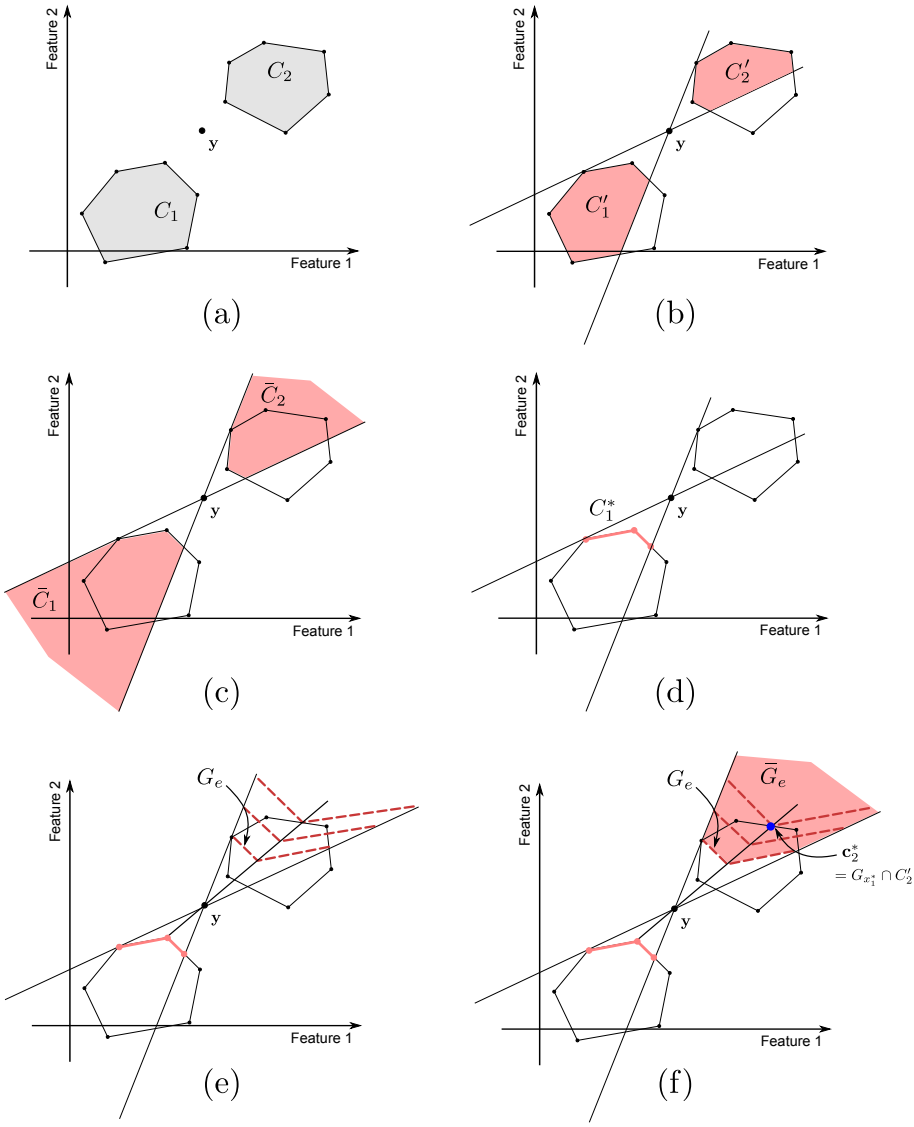


Figure 8.2: (a) An example of the sets C_1, C_2 and a mixture vector $\mathbf{y}$ for $q = 2$. (b) Illustration of the sets C_1' and C_2'. (c) Illustration of the sets $\bar{C}_1$ and $\bar{C}_2$. (d) Illustration of the set $\bar{C}_1^*$. (e) Illustration of the set G_e. (f) Illustration of the upper level set $\bar{G}_e$ of f_6 (at level e) and the *optimal source vector* $\mathbf{c}_2^*$.

This vector function is denoted[2]

$$\begin{aligned} \mathbf{g} : \bar{C}_2 &\to C_1^* \\ \mathbf{c}_2 &\mapsto \underline{(\mathbf{y}, \mathbf{c}_2)} \cap C_1^* \end{aligned}$$

As the definition of $\mathbf{g}$ assures that the bilinear constraints are always satisfied, the bilinear constraint can be left out of the optimization procedure. Additionally, as the objective function of $\mathcal{M}_5$ is automatically constrained to the unit interval, the bounds on x_1 can be dropped as well (in fact x_1 can be eliminated from the optimization problem). As such, we arrive at the following reformulation:

$$\begin{aligned} \mathcal{M}_6 : \quad & \underset{\mathbf{c}_2 \in \mathbb{R}^q}{\text{maximize}} \quad \frac{\|\mathbf{c}_2 - \mathbf{y}\|_2}{\|\mathbf{g}(\mathbf{c}_2) - \mathbf{y}\|_2 + \|\mathbf{c}_2 - \mathbf{y}\|_2} \\ & \text{subject to} \quad \mathbf{c}_2 \in C_2' \,. \end{aligned}$$

As we will be using the objective function of $\mathcal{M}_6$ several times in the remainder of this section, we denote this function as f_6.

We now show that the set of contour vectors of the objective function of $\mathcal{M}_6$, *i.e.* the vectors $\mathbf{c}_2$ for which $f_6(\mathbf{c}_2) = e$ (where $e \in [0, 1]$ is a constant), equals

$$G_e = \{\mathbf{c} \in \mathbb{R}^q \mid (\exists\, \mathbf{c}_1 \in C_1^*)\, (\mathbf{c} - \mathbf{y} = \frac{e}{e-1}\, (\mathbf{c}_1 - \mathbf{y}))\} \,.$$

For an illustration, see Figure 8.2(e). As before, we show this for $\mathbf{y} = \mathbf{0}_q$ (which can always be obtained by a translation). We start by noting that for any given $\mathbf{c}_2 \in C_2'$, there exists a $d < 0$ such that $\mathbf{g}(\mathbf{c}_2) = d\,\mathbf{c}_2$. On the other hand, for any $\mathbf{c}_2 \in C_2'$, we have that $\mathbf{c}_2 = d\,\mathbf{c}_1$ for some $\mathbf{c}_1 \in C_1^*$ and $d < 0$. As such, we can choose a vector $\dot{\mathbf{c}}_1 \in C_1^*$ and a scalar $t < 0$ and let $\dot{\mathbf{c}}_2 = t\,\dot{\mathbf{c}}_1$. This leads to the following equality $(\dot{\mathbf{c}}_1, \dot{\mathbf{c}}_2) = (\dot{\mathbf{c}}_1, t\,\dot{\mathbf{c}}_1) = (\mathbf{g}(\dot{\mathbf{c}}_2), \dot{\mathbf{c}}_2)$. We then have

$$f_6(\dot{\mathbf{c}}_2) = \frac{\|\dot{\mathbf{c}}_2\|}{\|\mathbf{g}(\dot{\mathbf{c}}_2)\| + \|\dot{\mathbf{c}}_2\|} = \frac{-t\,\|\dot{\mathbf{c}}_1\|}{\|\dot{\mathbf{c}}_1\| - t\,\|\dot{\mathbf{c}}_1\|} = \frac{-t}{1-t} \,. \tag{8.20}$$

The equality above shows that, for a fixed t and any $\mathbf{c}_2$ that can be written as $\mathbf{c}_2 = t\,\mathbf{c}_1$, where $\mathbf{c}_1 \in C_1^*$, we have $f_6(\mathbf{c}_2) = -t/(1-t)$. To obtain the contour $f_6(\mathbf{c}_2) = e$, we simply choose $t = \frac{e}{e-1}$. This shows that every vector in G_e has the same objective function value.

Any point $\mathbf{c}_2 \in \bar{C}_2$ can be written as $\mathbf{c}_2 = d\,\mathfrak{g}_e$, where $\mathfrak{g}_e \in G_e$ and $d > 0$. Moreover, there exists a unique $\mathbf{c}_1^* \in C_1^*$ such that $\mathfrak{g}_e = \frac{e}{e-1}\mathbf{c}_1^*$. We have that

[2] Strictly speaking, the function that is presented here maps a vector to a set. However, this set is always a singleton, and the unique element in this set is a q-dimensional vector. As a result, we identify this mapping with a vector function that maps an input to a q-dimensional vector.

$\mathbf{g}(\mathfrak{g}_e) = \mathbf{g}(\mathbf{c}_2) = \mathbf{c}_1^*$, leading to:

$$f_6(\mathbf{c}_2) = \frac{\|\mathbf{c}_2\|_2}{\|\mathbf{g}(\mathbf{c}_2)\|_2 + \|\mathbf{c}_2\|_2} = \frac{d\,\|\mathfrak{g}_e\|_2}{\|\mathbf{c}_1^*\|_2 + \|d\,\mathfrak{g}_e\|_2} = \frac{-d\frac{e}{e-1}}{1 - d\frac{e}{e-1}}\,.$$

For a point $\mathbf{c}_2 \in C_2'$ that can be written as $\mathbf{c}_2 = d\,\mathfrak{g}_e$ for some $d > 0$ and $\mathfrak{g}_e \in G_e$, we now have that

- If $d \in [0, 1[$, then $f_6(\mathbf{c}_2) < e$.
- If $d = 1$, then $f_6(\mathbf{c}_2) = e$.
- If $d \in]1, \infty[$, then $f_6(\mathbf{c}_2) > e$.

These implications show that if $\mathbf{c}_2 \in C_2' \setminus G_e$, then $f_6(\mathbf{c}_2) \neq e$, so G_e contains all contour vectors, and no others.

Let us now define the following set:

$$\bar{G}_e = \{\mathbf{c} \in \bar{C}_2 \mid (\exists\, \mathbf{c}_2 \in C_2')\,(\exists t \in [1, \infty[)\,(f_6(\mathbf{c}_2) = e \wedge \mathbf{c} = t\,\mathbf{c}_2)\}\,.$$

For an illustration, see Figure 8.2(f). The implications given above show that for any $\mathbf{c}_2 \in \bar{G}_e$, we have that $f_6(\mathbf{c}_2) \geq e$ (*i.e.* $\bar{G}_e$ is the upper level set of f_6). Moreover, the following equivalence holds:

$$\mathbf{c}_2 \in \bar{G}_e \iff (\exists\, \mathbf{c}_1 \in \bar{C}_1)\left(\mathbf{c}_2 = \frac{e}{e-1}\,\mathbf{c}_1\right)\,. \tag{8.21}$$

From this equivalence, we have that $\bar{G}_e$ can be seen as a point reflection of $\bar{C}_1$ through the origin followed by an isotropic scaling. As each of these operations preserves the convexity of a set, it can be concluded that $\bar{G}_e$ is a convex set. As such, $C_2' \cap \bar{G}_e$, which is the upper level set of the objective function on the feasible domain, is a convex set. This means that optimization problem $\mathcal{M}_6$ is a quasi-concave maximization problem (see Definition 3.14), having (except for some degenerate cases) one local (and thus global) optimal point.

The optimal value is attained by increasing e up to the point where $C_2' \cap \bar{G}_e$ is a singleton. For this case, we have that $e = \sup(X^1)$. Moreover, when the optimal point of $\mathcal{M}_4$ is denoted as $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$, we have that $\mathbf{c}_2^* = \bar{G}_{x_1^*} \cap C_2'$ (see Figure 8.2(f) for an illustration).

Conclusions

The reasoning above has three important implications:

- Firstly, instead of solving $\mathcal{M}_4$ directly, we can optimize an equivalent quasi-concave optimization problem (which leads to a series of convex feasibility problems). As such, it generalizes the approach proposed in the previous section, as this procedure does not require an initial feasible point.

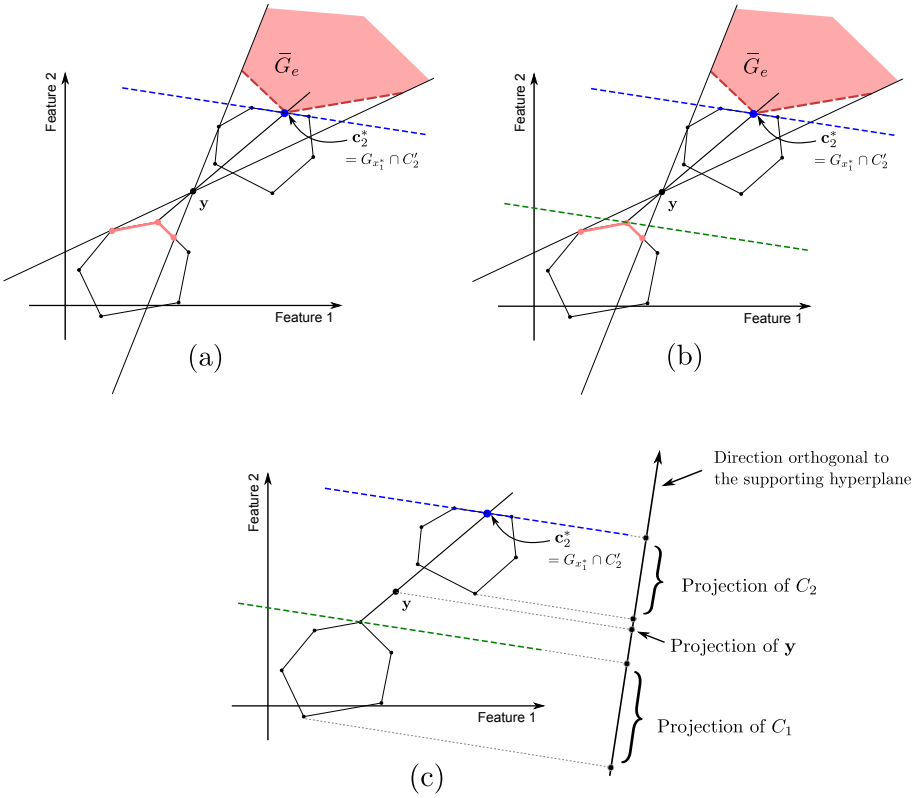


Figure 8.3: (a) An illustration (blue line) of the supporting hyperplane of the sets C_2 and $\bar{G}_{x_1^*}$ at $\mathbf{c}_2^*$. (b) An illustration (green line) of the supporting hyperplane of C_1 at $\mathbf{c}_2^*$ and parallel to the former hyperplane (blue line). (c) An illustration of the projection of C_1, C_2 and $\mathbf{y}$ on a direction that is orthogonal to the supporting hyperplanes.

- Secondly, it lays the foundations for a deeper characterization of the solution of $\mathcal{M}_4$:

  The optimal value occurs when e is chosen such that the upper level set $C_2' \cap \bar{G}_e$ is the singleton $\{\mathbf{c}_2^*\}$. For this value (*i.e.* $e = x_1^*$), we have that the convex set $\bar{G}_e$ touches C_2. This means that there exists at least one hyperplane passing through $C_2' \cap \bar{G}_e$ that is a supporting hyperplane (see Definition 3.10) for both C_2' and $\bar{G}_e$ at $\mathbf{c}_2^*$ (see Figure 8.3(a) for an illustration). Moreover, as $\bar{G}_e$ is a scaled point reflection of $\bar{C}_1$, there exists a hyperplane that is parallel to the former one, and is supporting for C_1' at $\underline{(\mathbf{c}_2^*, \mathbf{y})} \cap C_1^* = \{\mathbf{c}_1^*\}$ (see Figure 8.3(b) for an illustration).

- We can now (orthogonally) project C_1, C_2 and $\mathbf{y}$ on a line that is orthogonal to these hyperplanes (see Figure 8.3(c) for an illustration). Moreover, it is easy to see that when solving the optimization problem in this projected space, the optimal objective function value is identical to the original one. In the following section we will derive a procedure that allows to compute this direction very efficiently.

- Lastly, as a consequence of the previous item, in the special case that C_1 and C_2 are convex polytopes, we have that at least one of $\mathbf{c}_1^*$ and $\mathbf{c}_2^*$ is located at a vertex of C_1 or C_2 when $q = 2$.

8.4.4. Reformulation as a (linear) fractional program

In this section, we will show that $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ is equivalent to the following fractional program:

$$
\begin{aligned}
\mathcal{M}_7 : \quad & \underset{\mathbf{a}\in\mathbb{R}^q, b_1, b_2 \in \mathbb{R}}{\text{minimize}} && \frac{b_2 - \langle \mathbf{a}, \mathbf{y} \rangle}{b_2 - b_1} \\
& \text{subject to} && \|\mathbf{a}\|_2 = 1\,, \\
& && b_1 \geq \langle \mathbf{c}_1, \mathbf{a} \rangle\,, \qquad \text{for all}\, \mathbf{c}_1 \in C_1\,, \\
& && b_2 \geq \langle \mathbf{c}_2, \mathbf{a} \rangle\,, \qquad \text{for all}\, \mathbf{c}_2 \in C_2\,, \\
& && b_1 < \langle \mathbf{a}, \mathbf{y} \rangle\,, \\
& && b_2 > \langle \mathbf{a}, \mathbf{y} \rangle\,.
\end{aligned}
$$

The reasoning behind $\mathcal{M}_7$ is the following. When the sets C_1 and C_2 and the vector $\mathbf{y}$ are projected orthogonally onto the direction $\mathbf{a}$, $\mathcal{M}_4$ is transformed into a one-dimensional format. More precisely, given a unit vector $\mathbf{a} \in \mathbb{R}^q$, the coordinate of a point $\mathbf{d} \in \mathbb{R}^q$ on the axis that is defined by $\mathbf{a}$, can be computed (and denoted) as follows

$$P_{\mathbf{a}}(\mathbf{d}) = \langle \mathbf{a}, \mathbf{d} \rangle\,,$$

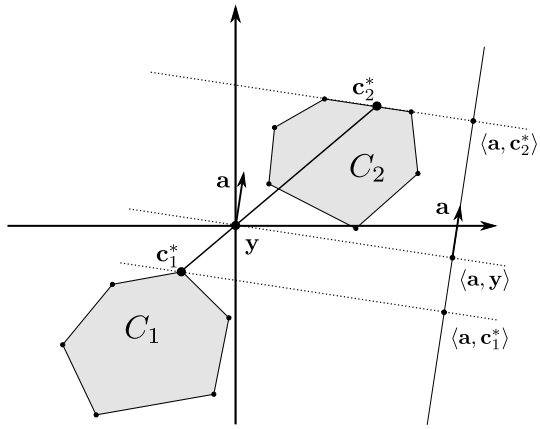


Figure 8.4: Derivation of a direction $\mathbf{a}$ such that $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ and $\mathcal{M}_4(P_{\mathbf{a}}(\mathbf{y}), P_{\mathbf{a}}(C_1), P_{\mathbf{a}}(C_2))$ are equivalent. Here, we let $\mathbf{y} = \mathbf{0}_2$.

moreover, the projection of a set $D \in \mathbb{R}^q$ on $\mathbf{a}$ is

$$P_{\mathbf{a}}(D) = \{\langle \mathbf{a}, \mathbf{d} \rangle \mid \mathbf{d} \in D\}.$$

This projection can be used to transform $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ into a new optimization problem $\mathcal{M}_4(P_{\mathbf{a}}(\mathbf{y}), P_{\mathbf{a}}(C_1), P_{\mathbf{a}}(C_2))$. Interestingly, the optimal value of $\mathcal{M}_4(P_{\mathbf{a}}(\mathbf{y}), P_{\mathbf{a}}(C_1), P_{\mathbf{a}}(C_2))$ is greater than or equal to the optimal value of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ (we show this hereafter). On the other hand, from the previous section, we know that there exists a direction such that the optimal values of both optimization problems are equal. In this perspective, $\mathcal{M}_7$ searches the direction $\mathbf{a}$ that minimizes the optimal value of $\mathcal{M}_4(P_{\mathbf{a}}(\mathbf{y}), P_{\mathbf{a}}(C_1), P_{\mathbf{a}}(C_2))$. We now formally prove that $\mathcal{M}_4$ and $\mathcal{M}_7$ are equivalent. The optimal direction is illustrated in Figure 8.4.

Proposition 8.10. *Let C_1 and C_2 be subsets of $\mathbb{R}^q$ such that $\mathcal{M}_4(\mathbf{0}_q, C_1, C_2)$ has feasible points and $\mathbf{0}_q \notin C_1$; then we have the following correspondence between the optimal point $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ of $\mathcal{M}_4(\mathbf{0}, C_1, C_2)$ and the optimal point $(\mathbf{a}^*, b_1^*, b_2^*)$ of $\mathcal{M}_7(\mathbf{0}, C_1, C_2)$:*

$$x_1^* = \frac{b_2^*}{b_2^* - b_1^*}.$$

Proof. From the discussion in Section 8.4.3, we know that there exists at least one pair of parallel hyperplanes such that the first hyperplane of this pair is a supporting hyperplane of C_1 at $\mathbf{c}_1^*$ and the second hyperplane of this pair is a supporting hyperplane of C_2 at $\mathbf{c}_2^*$. We now assume that $\mathbf{a}^*$ is the normal vector of these hyperplanes (we show later that this is assumption is correct).

We now have that

$$\frac{\langle \mathbf{a}^*, \mathbf{c}_2^* \rangle}{\langle \mathbf{a}^*, \mathbf{c}_2^* \rangle - \langle \mathbf{a}^*, \mathbf{c}_1^* \rangle} = x_1^* \,.$$

Moreover, we have that $\langle \mathbf{a}^*, \mathbf{c}_1 \rangle \leq \langle \mathbf{a}^*, \mathbf{c}_1^* \rangle$ for all $\mathbf{c}_1 \in C_1$, and $\langle \mathbf{a}^*, \mathbf{c}_2 \rangle \leq \langle \mathbf{a}^*, \mathbf{c}_2^* \rangle$ for all $\mathbf{c}_2 \in C_2$. This means that, for $\mathcal{M}_7$, the point $(\mathbf{a}^*, b_1, b_2)$ is feasible, with

$$b_1 = \langle \mathbf{a}^*, \mathbf{c}_1^* \rangle \,, \tag{8.22}$$

$$b_2 = \langle \mathbf{a}^*, \mathbf{c}_2^* \rangle \,. \tag{8.23}$$

The procedure described above allows, given $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$, for the construction of a point $(\mathbf{a}^*, b_1, b_2)$ that is feasible for $\mathcal{M}_7$, and moreover

$$x_1^* = \frac{b_2}{b_2 - b_1} \,.$$

We will now show that

(i) $\mathbf{a}^*$ is the normal vector of a pair of parallel supporting hyperplanes such that the first hyperplane of this pair is a supporting hyperplane of C_1 at $\mathbf{c}_1^*$ and the second hyperplane of this pair is a supporting hyperplane of C_2 at $\mathbf{c}_2^*$,

(ii) $b_1^* = \langle \mathbf{a}^*, \mathbf{c}_1^* \rangle$,

(iii) $b_2^* = \langle \mathbf{a}^*, \mathbf{c}_2^* \rangle$.

Firstly, when $\mathbf{a}$ is fixed (instead of an optimization variable for $\mathcal{M}_7$), it is easy to see that setting $b_1 = \max(P_{\mathbf{a}}(C_1))$ and $b_2 = \max(P_{\mathbf{a}}(C_2))$ optimizes $\mathcal{M}_7$. Moreover, when $\mathbf{a}$ is the normal vector of the supporting hyperplanes at $\mathbf{c}_1^*$ and $\mathbf{c}_2^*$, it is easy to see that $b_1 = \max(P_{\mathbf{a}}(C_1)) = \langle \mathbf{a}, \mathbf{c}_1^* \rangle$ and $b_2 = \max(P_{\mathbf{a}}(C_2)) = \langle \mathbf{a}, \mathbf{c}_2^* \rangle$. We now prove that $\mathbf{a}^*$ is the normal vector of these hyperplanes (and optimal for $\mathcal{M}_7$).

For any unit vector $\mathbf{a}$, it can easily be seen (as $\mathbf{c}_1^* = t\,\mathbf{c}_2^*$ for some $t \in \mathbb{R}$) that

$$\frac{\langle \mathbf{a}, \mathbf{c}_2^* \rangle}{\langle \mathbf{a}, \mathbf{c}_2^* \rangle - \langle \mathbf{a}, \mathbf{c}_1^* \rangle} = x_1^* \,.$$

Moreover, as $P_{\mathbf{a}}(\mathbf{c}_1^*) \in P_{\mathbf{a}}(C_1)$, we have that $P_{\mathbf{a}}(\mathbf{c}_1^*) \leq \max(P_{\mathbf{a}}(C_1))$ (and, equivalently, $P_{\mathbf{a}}(\mathbf{c}_2^*) \leq \max(P_{\mathbf{a}}(C_2))$). This means that, when $\mathbf{a}$ is feasible,

$$x_1^* = \frac{\langle \mathbf{a}, \mathbf{c}_2^* \rangle}{\langle \mathbf{a}, \mathbf{c}_2^* \rangle - \langle \mathbf{a}, \mathbf{c}_1^* \rangle} \leq \frac{\max(P_{\mathbf{a}}(C_2))}{\max(P_{\mathbf{a}}(C_2)) - \max(P_{\mathbf{a}}(C_1))} \,. \tag{8.24}$$

More precisely, when choosing $\mathbf{a}$ orthogonal to the supporting hyperplane of C_1 (resp. C_2) passing through $\mathbf{c}_1^*$ (resp. $\mathbf{c}_2^*$), inequality (8.24) becomes an equality. This completes the proof that the point in (i)–(iii) is a minimizer of $\mathcal{M}_7$.

□

With respect to the proposition above, it should be noted that, for $(\mathbf{a}^*, b_1^*, b_2^*)$ to be a strict local minimizer, there should exist exactly one pair of parallel supporting hyperplanes at $\mathbf{c}_1^*$ and $\mathbf{c}_2^*$. Indeed, if multiple pairs of parallel supporting hyperplanes exist, each of these pairs can be used to obtain a unit vector that turns (8.24) into an equality. Loosely speaking, we can say that if the boundary of C_1 in the neighborhood of $\mathbf{c}_1^*$ *or* C_2 in the neighborhood of $\mathbf{c}_2^*$ is smooth, then the local minimizer is strict.

From Proposition 8.10, we have the equivalence between $\mathcal{M}_4$ and $\mathcal{M}_7$. This means that $\mathcal{M}_7$ can be used as a substitute for $\mathcal{M}_4$. When dropping the norm constraint in $\mathcal{M}_7$, a fractional program $(\mathcal{M}_8)$ is obtained that is equivalent to $\mathcal{M}_7$ (however, implicitly we need that $\|\mathbf{a}\|_2 \neq 0$). Formally, we define $\mathcal{M}_8$ as:

$$
\begin{aligned}
\mathcal{M}_8 : \quad & \underset{\mathbf{a}\in\mathbb{R}_0^q, b_1, b_2\in\mathbb{R}}{\text{minimize}} && \frac{b_2 - \langle \mathbf{a}, \mathbf{y}\rangle}{b_2 - b_1} \\
& \text{subject to} && b_1 \geq \langle \mathbf{c}_1, \mathbf{a}\rangle \,, \qquad \text{for all}\, \mathbf{c}_1 \in C_1 \,, \\
& && b_2 \geq \langle \mathbf{c}_2, \mathbf{a}\rangle \,, \qquad \text{for all}\, \mathbf{c}_2 \in C_2 \,, \\
& && b_1 < \langle \mathbf{a}, \mathbf{y}\rangle \,, \\
& && b_2 > \langle \mathbf{a}, \mathbf{y}\rangle \,.
\end{aligned}
$$

In $\mathcal{M}_8$, the objective function is the ratio of two affine functions of b_1 and b_2. Moreover, the affine function in the denominator is strictly positive in the feasible domain. As such, the objective function of $\mathcal{M}_8$ is quasi-convex. All problem constraints are affine, making $\mathcal{M}_8$ a *potentially* tractable optimization problem. We say potentially, as the number of affine inequality constraints can be infinite. However, several measures can be taken to overcome this problem. We elaborate on these cases in the following section.

8.4.5. Solving fractional program $\mathcal{M}_8$

As a starting point of this section, consider the constraint(s):

$$
b_1 \geq \langle \mathbf{c}_1, \mathbf{a}\rangle \,, \quad \text{for all}\, \mathbf{c}_1 \in C_1 \,. \tag{8.25}
$$

As stated at the end of the previous section, the expression above potentially implies an infinite number of affine constraints. Fortunately, these constraints can easily be translated into a single constraint:

$$
b_1 \geq \sup_{\mathbf{c}_1 \in C_1} \langle \mathbf{c}_1, \mathbf{a}\rangle \,.
$$

These observations could suggest to use cutting plane algorithms [12] to solve $\mathcal{M}_8$.

Even though this approach potentially allows for very generic forms of C_1 and C_2 for which $\mathcal{M}_8$ can be solved efficiently, we will focus on two specific cases here that do not require the use of cutting plane algorithms.

- Notably, when C_1 and C_2 are convex polytopes, the constraints of $\mathcal{M}_8$ reduce to a finite set of linear inequalities. More precisely, the first set of constraints can be replaced with

$$\langle \mathbf{c}_1, \mathbf{a} \rangle \leq b_1 \,, \text{ for all vertices } \mathbf{c}_1 \text{ of } C_1 \,.$$

  In this case, $\mathcal{M}_8$ reduces to a linear fractional program. It is well known that linear fractional programs can be transformed into equivalent linear programs that can be solved efficiently[3].

- As a second interesting situation, we consider the case where C_1 and C_2 are ellipsoids. An ellipsoid in a q-dimensional space can be described by a q-vector and a positive definite $q \times q$ matrix. Therefore, let the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ as well as the positive definite matrices $\mathbf{V}^1$ and $\mathbf{V}^2$ be given such that:

$$\begin{aligned} C_1 &= \{\mathbf{v}_1 + \mathbf{V}^1\mathbf{u} \mid \mathbf{u} \in \mathbb{R}^q \wedge \|\mathbf{u}\|_2 \leq 1\} \,, \\ C_2 &= \{\mathbf{v}_2 + \mathbf{V}^2\mathbf{u} \mid \mathbf{u} \in \mathbb{R}^q \wedge \|\mathbf{u}\|_2 \leq 1\} \,. \end{aligned}$$

  It is well known that [10]

$$\max\{\mathbf{a}^\top \mathbf{c}_1 \mid \mathbf{c}_1 \in C_1\} = \mathbf{v}_1^\top \mathbf{a} + \left\|\mathbf{V}^1\mathbf{a}\right\|_2 \,.$$

  Therefore, constraint (8.25) reduces to the following second order cone constraint[4]:

$$\mathbf{v}_1^\top \mathbf{a} + \left\|\mathbf{V}^1\mathbf{a}\right\|_2 \leq b_1 \,.$$

  An equivalent procedure can be applied to C_2.

  Note that, when C_1 and C_2 are polytopes, we argued that the resulting optimization problem can be transformed into an equivalent linear programming problem. Fortunately, the presence of the second order cone constraint does not prevent us from applying a similar strategy here. Therefore, once this transformation is applied, second order cone programming solvers can be used to optimize the resulting program efficiently.

Conclusions

In this section, we proposed a reformulation of $\mathcal{M}_4$ that can be solved efficiently

[3] As we do not wish to disturb the flow of this chapter, we defer a description of this transformation to Appendix 8.A.

[4] This constraint is called a second order cone constraint as it can be written as a generalized inequality using the second order Lorentz cone.

with existing mathematical programming solvers. In the following chapter, we will be focusing on applying these solvers to compute X^k for a wide range of problem settings.

8.5. Robust set estimators

Up to this point, the problem *data* $\mathbf{y}$ and $(C_i)_{i=1}^n$ were assumed to be known exactly. However, in practice, there can be some uncertainty or variation on $\mathbf{y}$ and C_i. Such uncertainty can for instance be attributed to the presence of noise, or the imprecision that arises from rounding. In this section, we derive more robust set estimators. Firstly, we propose a set estimator that is robust w.r.t. $\mathbf{y}$. Secondly, we propose an estimator that is robust w.r.t. (estimates of) C_1 and C_2.

8.5.1. A y-robust estimator

To derive our $\mathbf{y}$-robust estimator, we will assume that $\mathbf{y}$ is given in an imprecise manner. More specifically, we will assume that $\mathbf{y}$ is an element of a given convex set $Y \subset \mathbb{R}^q$ (Figure 8.5.(a)). This contrasts the original setting, where $\mathbf{y}$ was assumed to be a fixed vector. In that perspective, the robust counterpart of the feasible set $X(\mathbf{y}, (C_i)_{i=1}^n)$ is now defined as

$$X(Y, (C_i)_{i=1}^n) = \left\{ \mathbf{x} \in \mathbb{S}^n \mid \left(\exists\, (\mathbf{c}_i)_{i=1}^n \in \bigtimes_{i=1}^n C_i \right) \left(\sum_{i=1}^n x_i\, \mathbf{c}_i \in Y \right) \right\} . \tag{8.26}$$

In the introduction, $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ was interpreted as the set of all *possible* values for the proportional contribution of the kth source to the mixture. Similarly, the set $X^k(Y, (C_i)_{i=1}^n)$ is defined as

$$X^k(Y, (C_i)_{i=1}^n) = \left\{ z \in \mathbb{R} \mid (\exists\, \mathbf{x} \in X^k(Y, (C_i)_{i=1}^n))(x_k = z) \right\} .$$

Here, $X^k(Y, (C_i)_{i=1}^n)$ is the set of all possible values for the proportional contribution of the kth source to an imprecisely defined mixture. It is easy to see that, for any $\mathbf{y} \in Y$, we have $X^k(\mathbf{y}, (C_i)_{i=1}^n) \subseteq X^k(Y, (C_i)_{i=1}^n)$. When Y is a convex set, $X^k(Y, (C_i)_{i=1}^n)$ is an interval. In the following section, we derive an optimization problem that can be used to compute $\sup(X^k(Y, (C_i)_{i=1}^n))$ efficiently. In what follows, we will generally assume that $k = 1$.

As a starting point, we define the mapping $h : (\mathbb{R}^q, 2^{\mathbb{R}^q}, 2^{\mathbb{R}^q}) \to \mathbb{R}$ as follows:

$$h(\mathbf{y}, C_1, C_2) = \begin{cases} \sup(X^1(\mathbf{y}, C_1, C_2)) \;, & \text{if } X^1(\mathbf{y}, C_1, C_2) \neq \emptyset \,, \\ -\infty \;, & \text{else.} \end{cases} \tag{8.27}$$

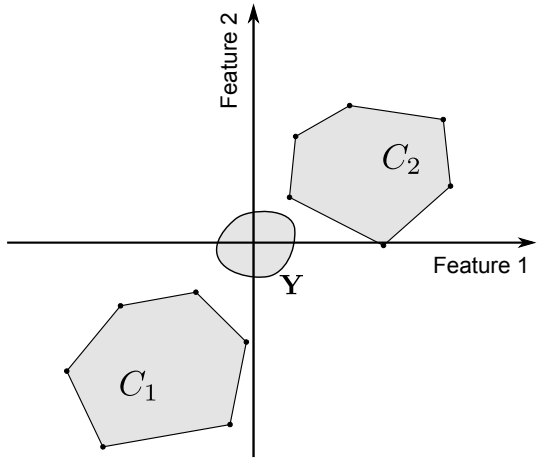


Figure 8.5: An illustration of a setting in which the mixture vector is described (imprecisely) by means of the set Y.

Using this function, we have

$$\sup(X^1(Y, C_1, C_2)) = \sup_{\mathbf{y} \in Y} h(\mathbf{y}, C_1, C_2)\,. \tag{8.28}$$

To compute this supremum, we distinguish three cases:

- Firstly, when $Y \cap C_1 \neq \emptyset$, we have the trivial solution $\sup(X^1(Y, C_1, C_2)) = 1$.

- Secondly, when $Y \cap \operatorname{conv}(C_1 \cup C_2) = \emptyset$, we have the trivial solution $\sup(X^1(Y, C_1, C_2)) = -\infty$ (*i.e.* the problem is unfeasible).

- Thirdly, when $Y \cap C_1 = \emptyset$ and $Y \cap \operatorname{conv}(C_1 \cup C_2) \neq \emptyset$, there exists a nontrivial supremum. Moreover, in this case, we have

$$\sup(X^1(Y, C_1, C_2)) = \sup_{\mathbf{y} \in Y'} h(\mathbf{y}, C_1, C_2)\,, \tag{8.29}$$

  where $Y' = Y \cap \operatorname{conv}(C_1 \cup C_2)$. Moreover, as we show next, h is a quasi-concave function of $\mathbf{y}$ when its domain is restricted to Y'.

Definition 8.3. *Given $\mathbf{y} \in \mathbb{R}^q$, $C_1 \in 2^{\mathbb{R}^q}$ and $C_2 \in 2^{\mathbb{R}^q}$, the vector $\mathbf{a} \in \mathbb{R}^q$ is called $\mathcal{M}_8(\mathbf{y}, C_1, C_2)$-feasible if there exists a pair of scalars b_1, b_2 such that $(\mathbf{a}, b_1, b_2)$ is a feasible point of $\mathcal{M}_8(\mathbf{y}, C_1, C_2)$.*

Lemma 8.11. *Consider a function $f : \mathbb{R} \to \mathbb{R}$ on the interval $[a, b]$ and two scalars*

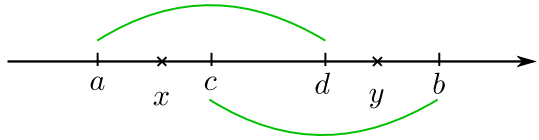


Figure 8.6: Illustration for the poof of Lemma 8.11.

$c, d \in]a, b[$ such that $c < d$. If

$$f \text{ is quasi-convex over the interval } [a, d], \tag{8.30}$$

$$f \text{ is quasi-convex over the interval } [c, b], \tag{8.31}$$

$$f(c) \neq f(d), \tag{8.32}$$

then f is quasi-convex over $[a, b]$.

Proof. Recall that a function $f : \mathbb{R} \to \mathbb{R}$ is quasi-convex over an interval $[\ell, r]$ if for any pair $c_1, c_2 \in [\ell, r]$ the following inequality holds:

$$f(\alpha\, c_1 + (1 - \alpha)\, c_2) \leq \max(f(c_1), f(c_2))\,, \qquad \text{for all } \alpha \in [0, 1]\,.$$

We now have the following property of a quasi-convex function: Given a function f that is quasi-convex over $[\ell, r]$ and two scalars $c_1 < c_2 \in [\ell, r]$ such that $f(c_1) < f(c_2)$. For any $c_3 \in [c_2, r]$, we have that $f(c_3) \geq f(c_2)$ (this can easily be proven by contradiction). Similarly, when $c_1 < c_2$ are such that $f(c_1) > f(c_2)$, then for any $c_3 \in [\ell, c_1]$, we have that $f(c_1) > f(c_2)$. We will use these properties later on.

We now use the results above to prove Lemma 8.11. Using the definition of a quasi-convex function, it suffices to prove that for any pair x, y where $x \in [a, c[$ and $y \in]d, b]$ (see Figure 8.6 for an accompanying illustration to this proof) we have that

$$f(\alpha\, x + (1 - \alpha)\, y) \leq \max(f(x), f(y))\,, \qquad \text{for all } \alpha \in [0, 1]\,.$$

Firstly, we will assume that $f(c) < f(d)$. We will now distinguish two options:

<u>Option 1: $f(x) \leq f(c)$</u>: Combining this option with the assumption that $f(c) < f(d)$ and the property of quasi-convex functions mentioned above, trivially leads to the following pair of inequalities:

$$f(d) \leq f(y)\,, \qquad\qquad f(x) < f(y)\,.$$

From this, we have that it suffices to prove that $f(z) \leq f(y)$ for any $z \in [x, y]$. We now have three options for z:

- $z \in [x, c]$: We have that $f(x) \leq f(c)$. As f is quasi-convex over $[a, d]$, we have that $f(z) \leq f(c)$, implying that $f(z) < f(y)$.

- $z \in [c, d]$: We have that $f(c) < f(d)$. As f is quasi-convex over $[a, d]$, we have that $f(z) \leq f(d)$, implying that $f(z) \leq f(y)$.
- $z \in [d, y]$: We have that $f(d) \leq f(y)$. As f is quasi-convex over $[c, b]$, we have that $f(z) \leq f(y)$.

Option 2: $f(x) > f(c)$: Combining this option with the assumption that $f(c) < f(d)$ and the property of quasi-convex functions mentioned above, trivially leads to the following inequality:

$$f(d) \leq f(y)\,.$$

We now prove that $f(z) \leq \max(f(x), f(y))$ for any $z \in [x, y]$. For z, we have the same three options as before. Moreover, only the proof for the first option differs from the previous case:

- $z \in [x, c]$: We have that $f(x) > f(c)$. As f is quasi-convex over $[a, d]$, we have that $f(z) < f(x)$.

The proof for the case $f(c) > f(d)$ is analogous.

□

The following lemma slightly generalizes Lemma 8.11.

Lemma 8.12. *Consider a function $f : \mathbb{R} \to \mathbb{R}$ on the interval $[a, b]$ and two scalars $c, d \in]a, b[$ such that $c < d$. If*

$$f \text{ is quasi-convex over the interval } [a, d], \tag{8.33}$$

$$f \text{ is quasi-convex over the interval } [c, b], \tag{8.34}$$

$$\exists\, e \in [c, d] \text{ such that } f(e) \neq f(c), \tag{8.35}$$

then f is quasi-convex over $[a, d]$.

Proof. This lemma can easily be proven by applying Lemma 8.11. As $e \in [c, d]$, we have that f is quasi-convex over the interval $[e, b]$. We now have that $[a, d]$ and $[e, b]$ are overlapping intervals such that $f(d) \neq f(e)$. From Lemma 8.11, it follows that f is quasi-convex over $[a, b]$. □

Corollary 8.13. *Consider a function $f : \mathbb{R} \to \mathbb{R}$ on the interval $[a, b]$ and two scalars $c, d \in]a, b[$ such that $c < d$. If*

$$f \text{ is quasi-concave over the interval } [a, d], \tag{8.36}$$

$$f \text{ is quasi-concave over the interval } [c, b], \tag{8.37}$$

$$\exists\, e \in [c, d] \text{ such that } f(e) \neq f(c), \tag{8.38}$$

then f is quasi-concave over $[a, d]$.

Proof. This corollary is a trivial consequence of Lemma 8.12. □

Proposition 8.14. *Let Y be a convex subset of $\mathbb{R}^2$, C_1 and C_2 be convex polytopes in $\mathbb{R}^2$ such that $X^1(Y, C_1, C_2) \neq \emptyset$ and $Y \cap C_1 = \emptyset$, then for any two vectors $\mathbf{y}_1, \mathbf{y}_2 \in Y \cap \operatorname{conv}(C_1 \cup C_2)$ the following holds:*

$$-h(\alpha\,\mathbf{y}_1 + (1-\alpha)\,\mathbf{y}_2, C_1, C_2) \leq \max(-h(\mathbf{y}_1, C_1, C_2), -h(\mathbf{y}_2, C_1, C_2))\,,$$

for each $\alpha \in [0,1]$ (in words, this means that h is a quasi-concave function of $\mathbf{y}$).

Proof. Let $Y' = Y \cap \operatorname{conv}(C_1 \cup C_2)$. For any $\mathbf{y} \in Y'$, it holds that $\sup(X^1(\mathbf{y}, C_1, C_2))$ is equal to the optimal value of $\mathcal{M}_8(\mathbf{y}, C_1, C_2)$. As a result, we can replace $\sup(X^1)$ with the optimal value of $\mathcal{M}_8$ in the definition of h (Eq. (8.27)). Additionally, for any $\mathbf{y} \in Y'$, we define $\mathbf{a}^*$ as the unit vector that optimizes $\mathcal{M}_8(\mathbf{y}, C_1, C_2)$ (provided that $\mathcal{M}_8$ is feasible). Finally, we note that, if $\mathbf{a}$ is $\mathcal{M}_8(\mathbf{y}, C_1, C_2)$-feasible, then the (sub)optimal value that can be obtained for $\mathcal{M}_8(\mathbf{y}, C_1, C_2)$ is

$$\frac{\max(P_{\mathbf{a}}(C_2)) - \langle \mathbf{a}, \mathbf{y} \rangle}{\max(P_{\mathbf{a}}(C_2)) - \max(P_{\mathbf{a}}(C_1))} \tag{8.39}$$

To keep the notation in the remainder of this proof uncluttered, we will drop C_1 and C_2 as arguments from h or $\mathcal{M}_8$ (as they remain constant throughout the proof).

We limit the proof to the case where $\mathbf{y}_1$ and $\mathbf{y}_2$ are chosen such that $h(\mathbf{y}_1) \leq h(\mathbf{y}_2)$ (the proof for the case where $h(\mathbf{y}_1) > h(\mathbf{y}_2)$ is similar). Moreover, we let $\mathbf{y}_3 = \alpha\mathbf{y}_1 + (1-\alpha)\mathbf{y}_2$ (for $\alpha \in [0,1]$).

We now consider two settings (1) $\mathbf{a}_3^*$ is both $\mathcal{M}_8(\mathbf{y}_1)$ and $\mathcal{M}_8(\mathbf{y}_2)$-feasible and (2) $\mathbf{a}_3^*$ is not $\mathcal{M}_8(\mathbf{y}_1)$-feasible or not $\mathcal{M}_8(\mathbf{y}_2)$-feasible.

Setting 1: $\mathbf{a}_3^*$ is both $\mathcal{M}_8(\mathbf{y}_1)$- and $\mathcal{M}_8(\mathbf{y}_2)$-feasible.

We now consider two cases:

- Case 1: $\langle \mathbf{a}_3^*, \mathbf{y}_1 \rangle \leq \langle \mathbf{a}_3^*, \mathbf{y}_2 \rangle$.

  From the definition of $\mathbf{y}_3$, it follows that

  $$\langle \mathbf{a}_3^*, \mathbf{y}_1 \rangle \leq \langle \mathbf{a}_3^*, \mathbf{y}_3 \rangle \leq \langle \mathbf{a}_3^*, \mathbf{y}_2 \rangle\,.$$

  Using these inequalities, we obtain

$$h(\mathbf{y}_3) = \frac{\max(P_{\mathbf{a}_3^*}(C_2)) - \langle \mathbf{a}_3^*, \mathbf{y}_3 \rangle}{\max(P_{\mathbf{a}_3^*}(C_2)) - \max(P_{\mathbf{a}_3^*}(C_1))} \geq \frac{\max(P_{\mathbf{a}_3^*}(C_2)) - \langle \mathbf{a}_3^*, \mathbf{y}_2 \rangle}{\max(P_{\mathbf{a}_3^*}(C_2)) - \max(P_{\mathbf{a}_3^*}(C_1))}\,.$$

Comparing this inequality with (8.39), it can be seen that (provided that $\mathbf{a}_3^*$ is feasible) the right hand side is a suboptimal value of $\mathcal{M}_8(\mathbf{y}_2)$. This means that the right-hand side is larger than $h(\mathbf{y}_2)$. Combining this with $h(\mathbf{y}_2) \geq h(\mathbf{y}_1)$, we have that $h(\mathbf{y}_3) \geq h(\mathbf{y}_2) \geq h(\mathbf{y}_1)$. This means that

$$-h(\mathbf{y}_3) \leq \max(-h(\mathbf{y}_1), -h(\mathbf{y}_2)) = -h(\mathbf{y}_1)\,,$$

which completes the proof for this case.

- Case 2: $\langle \mathbf{a}_3^*, \mathbf{y}_1 \rangle \geq \langle \mathbf{a}_3^*, \mathbf{y}_2 \rangle$.

  Similar arguments as before lead to

$$h(\mathbf{y}_3) = \frac{\max(P_{\mathbf{a}_3^*}(C_2)) - \langle \mathbf{a}_3^*, \mathbf{y}_3 \rangle}{\max(P_{\mathbf{a}_3^*}(C_2)) - \max(P_{\mathbf{a}_3^*}(C_1))} \geq \frac{\max(P_{\mathbf{a}_3^*}(C_2)) - \langle \mathbf{a}_3^*, \mathbf{y}_1 \rangle}{\max(P_{\mathbf{a}_3^*}(C_2)) - \max(P_{\mathbf{a}_3^*}(C_1))}\,.$$

  Here, the right-hand side is larger than $h(\mathbf{y}_1)$. Because of that, we have the following inequalities: $h(\mathbf{y}_1) \leq h(\mathbf{y}_3)$ and $h(\mathbf{y}_1) \leq h(\mathbf{y}_2)$. This means that

$$-h(\mathbf{y}_3) \leq \max(-h(\mathbf{y}_1), -h(\mathbf{y}_2)) = -h(\mathbf{y}_1)\,.$$

Setting 2: $\mathbf{a}_3^*$ is not $\mathcal{M}_8(\mathbf{y}_1)$-feasible or not $\mathcal{M}_8(\mathbf{y}_2)$-feasible.

We now generalize to the case where $\mathbf{a}_3^*$ is not $\mathcal{M}_8(\mathbf{y}_1)$-feasible or not $\mathcal{M}_8(\mathbf{y}_2)$-feasible. To make this generalization, we will explicitly assume that C_1 and C_2 are convex polytopes. Let us denote

$$\mathbf{y}^\alpha = \mathbf{y}_1 + \alpha(\mathbf{y}_2 - \mathbf{y}_1)\,. \tag{8.40}$$

In what follows, we will show that there always exist $\alpha_1 < \ldots < \alpha_{k+1} \in [0,1]$ with $\alpha_1 = 0$ and $\alpha_{k+1} = 1$ such that h is quasi-concave over $\overline{(\mathbf{y}^{\alpha_i}, \mathbf{y}^{\alpha_{i+2}})}$ $(i = 1, \ldots, k-1)$. Using the result in Corollary 8.13, we can then conclude that h is quasi-concave over $\overline{(\mathbf{y}_1, \mathbf{y}_2)}$.

(i) Let $\mathbf{y}$ be an element of the interior of $\mathrm{conv}(C_1 \cup C_2) \setminus C_1$, and denote the optimum of $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ as $(x_1^*, x_2^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$. From the discussion in Section 8.4.3, we have that at least one of $\mathbf{c}_1^*$ or $\mathbf{c}_2^*$ is a vertex of C_1 or C_2.

(ii) Without loss of generality, we assume that $\mathbf{c}_2^*$ is a vertex of C_2. Moreover, let $\mathbf{a}^*$ be the unit vector that optimizes $\mathcal{M}_8(\mathbf{y})$. Now consider the triangle $T_\mathbf{y}$ with top $\mathbf{c}_2^*$ and as base the line segment of C_1^* that contains $\mathbf{c}_1^*$ (see Figure 8.7(a) for an illustration). We know that $\mathbf{a}^*$ is orthogonal to the base of $T_\mathbf{y}$. It follows that for each $\mathbf{y}' \in T_\mathbf{y}$, $\mathbf{a}^*$ is optimal for $\mathcal{M}_8(\mathbf{y}')$. As a consequence, the contour lines of h in $T_\mathbf{y}$ are parallel to the base of $T_\mathbf{y}$ (see Figure 8.7(b) for an illustration).

(iii) Let $\mathbf{y}_1$ and $\mathbf{y}_2$ be two vectors such that $T_{\mathbf{y}_1} \neq T_{\mathbf{y}_2}$ have one line segment

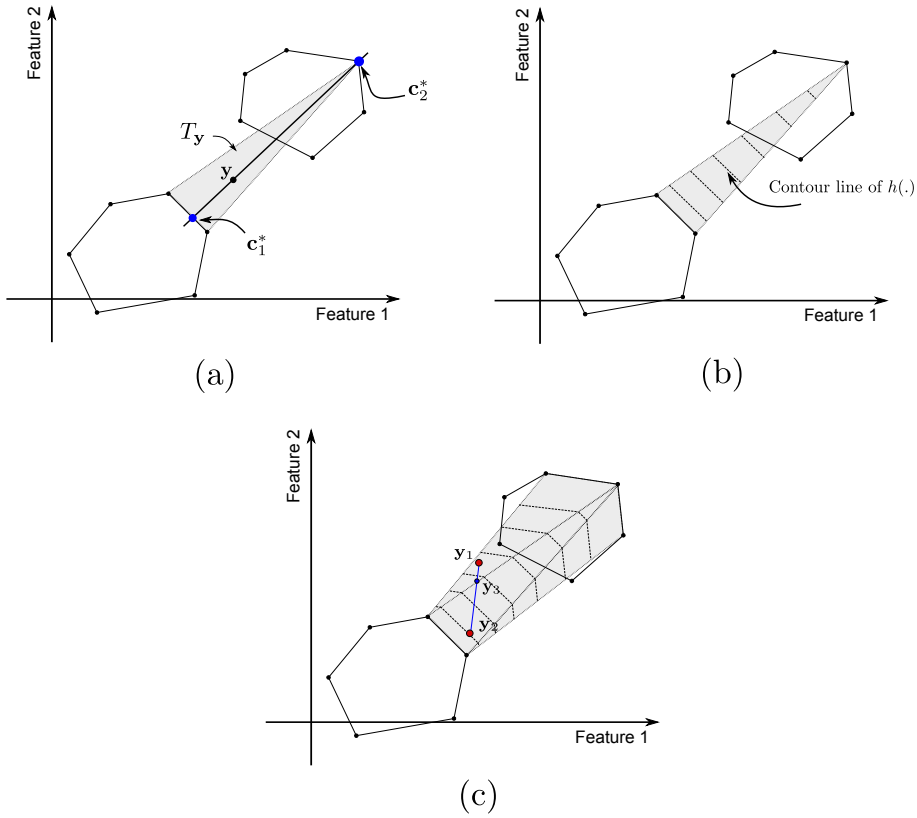


Figure 8.7: (a) Illustration of the triangle $T_{\mathbf{y}}$. (b) Illustration of the contour lines of h. (c) Illustration of a setting in which $T_{\mathbf{y}_1} \cap T_{\mathbf{y}_2}$ is a line segment. $\mathbf{y}_3$ is located at the intersection of that line segment with $\overline{(\mathbf{y}_1, \mathbf{y}_2)}$.

in common. It follows that $\overline{(\mathbf{y}_1, \mathbf{y}_2)} \subset T_{\mathbf{y}_1} \cup T_{\mathbf{y}_2}$. Finally, let $\{\mathbf{y}_3\} = T_{\mathbf{y}_1} \cap T_{\mathbf{y}_2} \cap \overline{(\mathbf{y}_1, \mathbf{y}_2)}$ and let $\mathbf{a}_3^*$ be a unit vector that optimizes $\mathcal{M}_8(\mathbf{y}_3)$ (see Figure 8.7(c) for an illustration). As $\mathbf{y}_3$ is an element of $T_{\mathbf{y}_1} \cap T_{\mathbf{y}_2}$, we know that $\mathbf{a}_1^*$ is $\mathcal{M}_8(\mathbf{y}_3)$-feasible and that $\mathbf{a}_2^*$ is $\mathcal{M}_8(\mathbf{y}_3)$-feasible. Therefore, from Setting 1, we have that h is quasi-concave over $\overline{(\mathbf{y}_1, \mathbf{y}_3)}$ and $\overline{(\mathbf{y}_3, \mathbf{y}_2)}$.

(iv) From (ii), it follows that both $\mathbf{a}_1^*$ and $\mathbf{a}_2^*$ are optimal for $\mathcal{M}_8(\mathbf{y}_3)$. Let us select $\mathbf{a}_3^* = \mathbf{a}_1^*$, implying that $\mathbf{a}_1^*$ is $\mathcal{M}_8(\mathbf{y}_3)$-feasible. Clearly, $\mathbf{a}_1^*$ is not necessarily $\mathcal{M}_8(\mathbf{y}_2)$-feasible. However, we now show that there always exists an $\epsilon \in]0, 1]$ such that $\mathbf{a}_3^*$ is $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_2 - \mathbf{y}_3))$- and $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3))$-feasible (see Figure 8.8(a) for an illustration).

(a) As $\mathbf{a}_3^* = \mathbf{a}_1^*$ and $\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3) \in T_{\mathbf{y}_1}$ for any $\epsilon \in [0, 1]$, we have from (ii) that $\mathbf{a}_3^*$ is $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3))$-feasible for any $\epsilon \in [0, 1]$. Moreover, $\mathbf{a}_3^*$ is optimal for $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3))$.

(b) As $\mathbf{a}_3^*$ is optimal for $\mathcal{M}_8(\mathbf{y}_3)$, it holds that:

$$\max(P_{\mathbf{a}_3^*}(C_1)) < \langle \mathbf{a}_3^*, \mathbf{y}_3 \rangle ,$$
$$\max(P_{\mathbf{a}_3^*}(C_2)) > \langle \mathbf{a}_3^*, \mathbf{y}_3 \rangle .$$

Moreover, we have that

$$\langle \mathbf{a}_3^*, \mathbf{y}_3 + \epsilon(\mathbf{y}_2 - \mathbf{y}_3) \rangle = \langle \mathbf{a}_3^*, \mathbf{y}_3 \rangle + \langle \mathbf{a}_3^*, \epsilon(\mathbf{y}_2 - \mathbf{y}_3) \rangle .$$

As the inequalities above are strict, it holds that when choosing $\epsilon]0, 1]$ sufficiently small, the following inequalities hold:

$$\max(P_{\mathbf{a}_3^*}(C_1)) < \langle \mathbf{a}_3^*, \mathbf{y}_3 + \epsilon\,(\mathbf{y}_2 - \mathbf{y}_3) \rangle , \tag{8.41}$$
$$\max(P_{\mathbf{a}_3^*}(C_2)) > \langle \mathbf{a}_3^*, \mathbf{y}_3 + \epsilon\,(\mathbf{y}_2 - \mathbf{y}_3) \rangle . \tag{8.42}$$

The means that $\mathbf{a}_3^*$ is $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_2 - \mathbf{y}_3))$-feasible.

As $\mathbf{a}_3^*$ is optimal for $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3))$ and $\mathbf{a}_3^*$ is $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_2 - \mathbf{y}_3))$-feasible, h we have from Setting 1 that h is quasi-concave over $\overline{(\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3), \mathbf{y}_3 + \epsilon(\mathbf{y}_2 - \mathbf{y}_3))}$.

(v) Additionally, we have that $h(\mathbf{y}_3 + \epsilon(\mathbf{y}_2 - \mathbf{y}_3)) \neq h(\mathbf{y}_3)$ or $h(\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3)) \neq h(\mathbf{y}_3)$. Combining this with (iii) and (iv), we have from Lemma 8.12 that h is quasi-concave over $\overline{(\mathbf{y}_1, \mathbf{y}_2)}$.

We can now extend the reasoning above to the case where $T_{\mathbf{y}_1}$ and $T_{\mathbf{y}_2}$ do not have a line segment in common (see Figure 8.8(b) for an illustration). Clearly, there always exists a set of triangles $\{T_{\mathbf{y}_i}\}_{i=1}^k$ such that $\overline{(\mathbf{y}_1, \mathbf{y}_2)} \subset \bigcup_{i=1}^k T_{\mathbf{y}_i}$. Let $\alpha_2 < \ldots < \alpha_k \in]0, 1[$ such that $\mathbf{y}^{\alpha_j}$ (defined using (8.40)) with $j = 2, \ldots, k$, correspond to the intersections of the triangles in $\{T_{\mathbf{y}_i}\}_{i=1}^k$ with line segment $\overline{(\mathbf{y}_1, \mathbf{y}_2)}$ (see Figure 8.8(b) for an illustration). Lastly, let $\alpha_1 = 0$ and $\alpha_{k+1} = 0$.

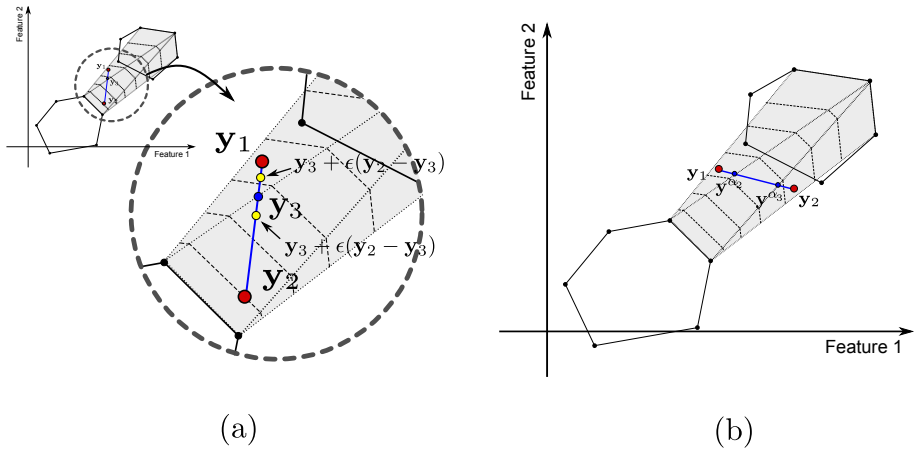


Figure 8.8: (a) Illustration item (iii) of the proof of Proposition 8.14. ϵ can be choosen such that a_3^* is $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_2 - \mathbf{y}_3))$ and $\mathcal{M}_8(\mathbf{y}_3 + \epsilon(\mathbf{y}_1 - \mathbf{y}_3))$-feasible. (b) Illustration of a setting in which $T_{\mathbf{y}_1}$ and $T_{\mathbf{y}_2}$ do not have a line segment in common.

It is clear that the line segments $\overline{(\mathbf{y}^{\alpha_i}, \mathbf{y}^{\alpha_{i+2}})}$ with $i = 1, \ldots, k-1$ are overlapping line segments. Moreover, applying (i)–(v), we know that h is quasi-concave over each of these line segments. As a result, h is quasi-concave over $\overline{(\mathbf{y}_1, \mathbf{y}_2)}$.

□

8.5.2. A C_i-robust estimator

It is clear that the sets $C_1, \ldots, C_n$ have an important influence on the set $X(\mathbf{y}, (C_i)_{i=1}^n)$. The examples in the previous chapter suggest that defining $C_1, \ldots, C_n$ as the convex hulls of sets of observed prototype vectors provides an interesting special case. Let us now formally assume that, for the ith source, we are given m_i data vectors $\mathbf{v}_1^i, \ldots, \mathbf{v}_{m_i}^i$. Mathematically, we now can define C_i as

$$C_i = \text{conv}\left(\{\mathbf{v}_1^i, \ldots, \mathbf{v}_{m_i}^i\}\right) . \tag{8.43}$$

We will now generalize our estimator to the case where $\mathbf{v}_1^i, \ldots, \mathbf{v}_{m_i}^i$ are defined imprecisely. More specifically, we will assume that the exact value of $\mathbf{v}_j^i$ is unknown; instead, we have that $\mathbf{v}_j^i \in V_j^i$, where V_j^i is a convex set (Figure 8.9(a)). These sets can now be used to create robust estimators of $\inf(X^k(\mathbf{y}, C_1, C_2))$ and $\sup(X^k(\mathbf{y}, C_1, C_2))$.

Depending on the semantics of the sets V_j^i, several 'robust' estimators can be defined. In this section, we interpret the sets V_j^i as a collection of sets that are known to contain the 'true' data vectors $\mathbf{v}_j^i$. This means that V_j^i are imprecise

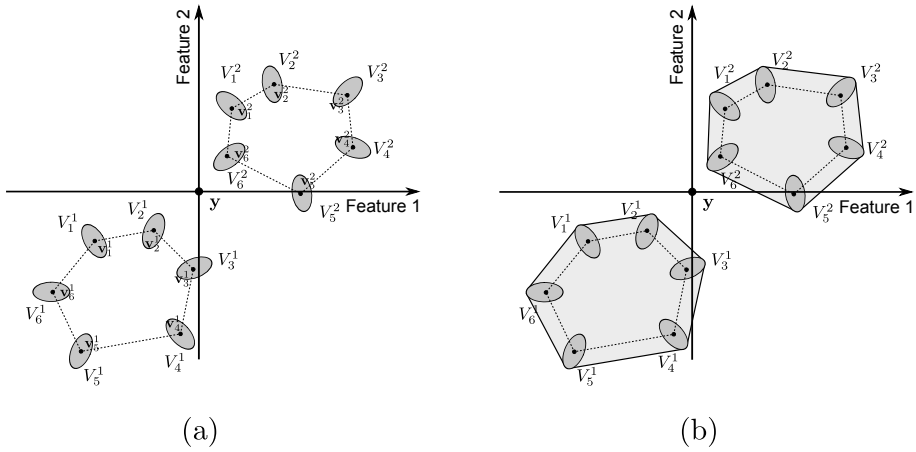


Figure 8.9: (a) A visualization of the observed prototype vectors $\mathbf{v}_j^1$ and $\mathbf{v}_j^2$, and the sets V_j^1 and V_j^2 (with $j = 1, \ldots, 6$) that can be used to describe prototypes in an imprecise manner. (b) The convex hulls of $\{V_j^1\}_{j=1}^6$ and $\{V_j^2\}_{j=1}^6$.

observations of these data vectors. This imprecision can be translated into a robust version of $\sup(X^k(\mathbf{y}, (C_i)_{i=1}^n))$ in two manners.

- Firstly, in a conservative manner, we could argue that our estimator should be an upper bound on $\sup(X^k(\mathbf{y}, (C_i)_{i=1}^n))$. Such an upper bound is defined as:

$$u = \max_{\mathbf{d}_j^i \in V_j^i} \left(\sup \left(X^k \left(\mathbf{y}, (\mathrm{conv}(\{\mathbf{d}_j^i\}_{j=1}^{m_i}))_{i=1}^n \right) \right) \right).$$

  It is not hard to see that (for an illustration see Figure 8.9(b))

$$u = \sup \left(X^k \left(\mathbf{y}, (\mathrm{conv}(\{V_j^i\}_{j=1}^{m_i}))_{i=1}^n \right) \right).$$

- Secondly, in an optimistic manner, we could as well try to find a lower bound on $\sup(X^k(\mathbf{y}, (C_i)_{i=1}^n))$:

$$\ell = \min_{\mathbf{d}_j^i \in V_j^i} \left(\sup \left(X^k \left(\mathbf{y}, (\mathrm{conv}(\{\mathbf{d}_j^i\}_{j=1}^{m_i}))_{i=1}^n \right) \right) \right).$$

In any case, we have that $\sup(X^k(\mathbf{y}, (C_i)_{i=1}^n)) \in [\ell, u]$.

It can easily be seen that (here for two classes) these bounds can be obtained by the following optimization problems (which are based on $\mathcal{M}_8$):

To obtain the upper bound, we can solve:

$$\begin{aligned}
\mathcal{M}_9 : \quad & \underset{\mathbf{a}\in\mathbb{R}_0^q, b_1, b_2\in\mathbb{R}}{\text{minimize}} && \frac{b_2 - \langle \mathbf{a}, \mathbf{y}\rangle}{b_2 - b_1} \\
& \text{subject to} && b_1 \geq \max(P_{\mathbf{a}}(V_j^1))\,, \qquad \text{for } j = 1, \dots m_1\,, \\
& && b_2 \geq \max(P_{\mathbf{a}}(V_j^2))\,, \qquad \text{for } j = 1, \dots m_2\,, \\
& && b_1 < \langle \mathbf{a}, \mathbf{y}\rangle\,, \\
& && b_2 > \langle \mathbf{a}, \mathbf{y}\rangle\,.
\end{aligned}$$

To obtain the lower bound, we can solve:

$$\begin{aligned}
\mathcal{M}_{10} : \quad & \underset{\mathbf{a}\in\mathbb{R}_0^q, b_1, b_2\in\mathbb{R}}{\text{minimize}} && \frac{b_2 - \langle \mathbf{a}, \mathbf{y}\rangle}{b_2 - b_1} \\
& \text{subject to} && b_1 \geq \min(P_{\mathbf{a}}(V_j^1))\,, \qquad \text{for } j = 1, \dots m_1\,, \\
& && b_2 \geq \min(P_{\mathbf{a}}(V_j^2))\,, \qquad \text{for } j = 1, \dots m_2\,, \\
& && b_1 < \langle \mathbf{a}, \mathbf{y}\rangle\,, \\
& && b_2 > \langle \mathbf{a}, \mathbf{y}\rangle\,.
\end{aligned}$$

Similar to the ellipsoidal sets in Section 8.4.5, a computationally interesting case arises when V_j^1 and V_j^2 are ellipsoids. More precisely, let

$$V_j^1 = \{\bar{\mathbf{v}}_j^1 + \mathbf{V}_j^1\mathbf{u} \mid \mathbf{u} \in \mathbb{R}^q \wedge \|\mathbf{u}\|_2 \leq 1\}\,,$$

where $\bar{\mathbf{v}}_j^i \in \mathbb{R}^q$ and $\mathbf{V}_j^1 \in \mathbb{R}^{q\times q}$ are fixed.

We now have that

$$\max\{\mathbf{a}^\top\mathbf{v} \mid \mathbf{v} \in V_j^1\} = \bar{\mathbf{v}}_j^{1\top}\mathbf{a} + \left\|\mathbf{V}_j^1\mathbf{a}\right\|_2 \leq b_1\,.$$

This reduces the constraints to a finite number of second-order cone constraints.

Unfortunately, the constraints in the optimization problem for the lower bound do not lead to second order cone constraints. Moreover, it is not difficult to show that this optimization problem is not (quasi-) convex and probably a hard problem to optimize.

8.6. Noisy observations

8.6.1. The LMM with a noisy observation of y

In the previous sections, it is assumed that the sets C_1 and C_2 and the mixture vector $\mathbf{y}$ are given. Moreover, the noisefree linear mixture model (7.1) was used to define $X(\mathbf{y}, (C_i)_{i=1}^2)$. However, in practice, we will often be dealing with a noisy observation $\tilde{\mathbf{y}}$ of the mixture vector $\mathbf{y}$. In that case, a mixture model that is capable of modeling the error that is associated with $\mathbf{y}$ may lead to more informative results. Consider the following probabilistic linear mixture model:

$$\tilde{\mathcal{Y}} = \underbrace{\sum_{i=1}^{n} x_i \, \mathbf{c}_i}_{\mathbf{y}} + \mathcal{E} \,, \tag{8.44}$$

where $\mathcal{E}$ is a $q \times 1$ random vector of error terms. Moreover, we assume that $\mathbb{E}[\mathcal{E}] = \mathbf{0}_q$. The random vector $\tilde{\mathcal{Y}}$ models a noisy version of the mixture vector $\mathbf{y}$. In the remainder of this section, we will assume that $\tilde{\mathbf{y}}$ is an observation of $\tilde{\mathcal{Y}}$.

It is clear that, in general, we have that $X^k(\tilde{\mathbf{y}}, C_1, C_2) \neq X^k(\mathbf{y}, C_1, C_2)$. In the remainder of this section, we present a procedure that can be used to construct an interval for which we can show that the probability that $X^k(\mathbf{y}, C_1, C_2)$ is a subset of this interval is bounded from below (at a controllable level).

Let $r \in]0, +\infty[$ and $\mathbf{h} \in \mathbb{R}^q$, we now define a hypercube that is centered at $\mathbf{h}$ and has a side with length r:

$$H(r, \mathbf{h}) = \{r \, \mathbf{d} + \mathbf{h} \mid \mathbf{d} \in [-1/2, 1/2]^q\} \,.$$

Moreover, given $\alpha \in [0, 1]$, let $r_\alpha > 0$ be a choosen such that

$$\Pr(\mathcal{E} \in H(r_\alpha, \mathbf{0}_q)) = \alpha \,.$$

For example, when $\mathcal{E}$ is multivariate normally distributed with mean vector $\mathbf{0}_q$ and known covariance matrix, the equicoordinate quantile function of Genz and Bretz [112] can be used to compute r_α.

As $\tilde{\mathcal{Y}} = \mathbf{y} + \mathcal{E}$, it is not hard to show that $\Pr(\mathbf{y} \in H(r_\alpha, \tilde{\mathcal{Y}})) = \alpha$. We can now use $H(r_\alpha, \tilde{\mathcal{Y}})$ to define the random sets $X(H(r_\alpha, \tilde{\mathcal{Y}}), C_1, C_2)$ and $X^k(H(r_\alpha, \tilde{\mathcal{Y}}), C_1, C_2)$. Moreover, we can give the following probabilistic interpretation to these sets:

$$\Pr\Big(X^k(\mathbf{y}, C_1, C_2) \subseteq X^k(H(r_\alpha, \tilde{\mathcal{Y}}), C_1, C_2)\Big) \geq \alpha \,.$$

In words, this means that the random interval $X^k(H(r_\alpha, \tilde{\mathcal{Y}}), C_1, C_2)$ will contain $X^k(\mathbf{y}, C_1, C_2)$ with a probability of at least α. Lastly, to compute the estimator above in practice for an observation $\tilde{\mathbf{y}}$ of $\tilde{\mathcal{Y}}$, the insights of Section 8.5.1 can be useful. More precisely, as $H(r_\alpha, \tilde{\mathbf{y}})$ is a convex subset of $\mathbb{R}^q$, the results in Section 8.5.1 can be used to develop an optimization problem to compute $\sup(X^k(H(r_\alpha, \tilde{\mathbf{y}}), C_1, C_2))$ efficiently.

8.6.2. The LMM with a noisy observation of $\mathbf{v}_i^j$

In the same spirit as the previous section, in many practical situations, the sets C_1 and C_2 are unknown but rather need to be estimated from data (we assume $\mathbf{y}$ is known exactly). We will simplify the discussion here by assuming that $C_1 = \text{conv}(\{\mathbf{v}_1^1, \ldots, \mathbf{v}_{m_1}^1\})$ and $C_2 = \text{conv}(\{\mathbf{v}_1^2, \ldots, \mathbf{v}_{m_2}^2\})$, where $\mathbf{v}_i^1$ and $\mathbf{v}_i^2$ are observed prototype vectors. However, instead of $\mathbf{v}_i^j$, we have been given an observation $\tilde{\mathbf{v}}_i^j$ of the random vector $\tilde{\mathcal{V}}_i^j$:

$$\tilde{\mathcal{V}}_i^j = \mathbf{v}_i^j + \mathcal{E}_i^j \,,$$

where $\mathcal{E}_i^j$ is a $q \times 1$ random vector of error terms. Moreover, we assume that $\mathbb{E}[\mathcal{E}_i^j] = \mathbf{0}_q$ and $\text{cov}(\mathcal{E}_i^j) = \Sigma$.

Given $\beta \in [0, 1]$, let $r_\beta > 0$ be a choosen such that

$$\Pr(\mathcal{E}_i^j \in H(r_\beta, \mathbf{0}_q)) = \sqrt[m]{\beta} \,,$$

where $m = m_1 + m_2$. It is not hard to show that $\Pr(\mathbf{v}_i^j \in H(r_\beta, \tilde{\mathcal{V}}_i^j)) = \sqrt[m]{\beta}$. Let us now define the random sets $\tilde{\mathcal{C}}_1^\beta$ and $\tilde{\mathcal{C}}_2^\beta$:

$$\tilde{\mathcal{C}}_1^\beta = \text{conv}(\{H(r_\beta, \tilde{\mathcal{V}}_i^1)\}_{i=1}^{m_1}) \quad \text{and} \quad \tilde{\mathcal{C}}_2^\beta = \text{conv}(\{H(r_\beta, \tilde{\mathcal{V}}_i^2)\}_{i=1}^{m_2}) \,.$$

It is not hard to see that $\Pr(C_1 \subseteq \tilde{\mathcal{C}}_1^\beta \text{ and } C_2 \subseteq \tilde{\mathcal{C}}_1^\beta) \geq \beta$. As a result, we have that

$$\Pr(X^k(\mathbf{y}, C_1, C_2) \subseteq X^k(\mathbf{y}, \tilde{\mathcal{C}}_1^\beta, \tilde{\mathcal{C}}_2^\beta)) \geq \beta \,.$$

Given two sets of noisy observed prototype vectors $\{\tilde{\mathbf{v}}_1^1, \ldots, \tilde{\mathbf{v}}_{m_1}^1\}$ and $\{\tilde{\mathbf{v}}_1^2, \ldots, \tilde{\mathbf{v}}_{m_2}^2\}$ we can efficiently compute

$$\sup(X^k(\mathbf{y}, \text{conv}(\{H(r_\beta, \tilde{\mathbf{v}}_i^1)\}_{i=1}^{m_1}), \text{conv}(\{H(r_\beta, \tilde{\mathbf{v}}_i^2)\}_{i=1}^{m_2})))$$

using the methodology described in Section 8.5.2. Lastly, it should be noted that the bound that is reported here is likely to be extremely conservative.

8.6.3. High-dimensional problems and non-informative dimensions

Up to this point, the dimensionality q of the space in which the sources and the mixture were represented (by means of the sets C_1 C_2 and the vector $\mathbf{y}$) was fixed. By increasing the dimensionality of the representation space, the description of the sources becomes more informative, and from the definition of $X(\mathbf{y}, C_1, C_2)$ and $X^k(\mathbf{y}, C_1, C_2)$, it can easily be seen that this increased dimensionality will lead to a reduction of the size of both $X(\mathbf{y}, C_1, C_2)$ and $X^k(\mathbf{y}, C_1, C_2)$. In most cases this reduction is interesting as it will reduce the uncertainty associated with our estimator.

When a representation space is extended with a new feature (let the sets $C_1^{\uparrow}, C_2^{\uparrow} \in \mathbb{R}^{q+1}$ and the vector $\mathbf{y}^{\uparrow} \in \mathbb{R}^{q+1}$ be the representations of the sources and the mixture vector in the extended space), we say that this new dimension is *non-informative* if

$$X(\mathbf{y}, C_1, C_2) = X(\mathbf{y}^{\uparrow}, C_1^{\uparrow}, C_2^{\uparrow})$$

for all $\mathbf{y}$ (and $\mathbf{y}^{\uparrow}$). For example, when a new feature is a copy of an existing feature, the new feature will be non-informative. Another example is the case where a new feature can be written as a function (for instance a linear combination) of features that are already present. Based on this discussion, we may conclude that non-informative features are neither helpful nor harmful for our problem. However, when a non-informative feature is measured in a noisy setting, it will lead to a reduction in the size of $X^k(\mathbf{y}^{\uparrow}, C_1^{\uparrow}, C_2^{\uparrow})$. Especially when the number of noisy non-informative features is high, this may lead to intervals that are overly narrow. In those cases we may consider using dimensionality reduction techniques that can recover a limited number of informative (latent) features. We can for instance use dimensionality reduction techniques such as (robust) principal component analysis. The assumption here is that the variability of the (noisy) observed prototype vectors $\tilde{\mathbf{v}}_i^j$ will be high in the *informative* directions and low in all other directions. Unfortunately, both the choice of a particular dimensionality reduction technique, and the dimensionality of the projected space, remain rather arbitrary.

8.7. Sparse solutions

Up to this point, $C_1, \ldots, C_n$ were assumed to be convex sets. In this section, we will relax this assumption. More precisely, we will assume that C_i is the union of multiple (elliptical) convex sets. We start this section by motivating why this case can be interesting to consider in a data analysis setting.

8.7.1. Why do we need sparse solutions?

As argued before, in several applications, the sets $C_1, \ldots, C_n$ are defined as the convex hulls of observed prototype vectors $\mathbf{v}_i^j$. Moreover, to mitigate the influence of noise (or imprecision) on the observations, we proposed a relaxation of the original problem by replacing each $\mathbf{v}_i^j$ by a set V_i^j (where V_i^j is for instance a ball centered at $\mathbf{v}_i^j$). Such a relaxation extends the range of X^k. In some cases, the intervals that are obtained in this manner can be rather wide. A natural way of reducing the range of X^k exists of reducing the size of the sets $C_1, \ldots, C_n$ (*i.e.* not defining them as the convex hulls of V_i^j). Several procedures can be used to reduce the size of these sets. However, the resulting sets should still be interpretable. An interesting case can be obtained by defining $C_1, \ldots, C_n$ as follows:

$$C_1 = \bigcup_{i=1}^{m_1} V_i^1 \,, \qquad \ldots \,, \qquad C_n = \bigcup_{i=1}^{m_n} V_i^n \,. \tag{8.45}$$

When V_i^j is a ball centered at $\mathbf{v}_i^j$ (which can for instance be an observed prototype vector), this formulation will imply that $\mathbf{c}_k$ is located near one of the observed prototype vectors $\{\mathbf{v}_i^k\}_{i=1}^{m_k}$. In a two-class case (*i.e.* $n = 2$, see Figure 8.10) this would imply that a mixture (with mixture vector $\mathbf{y}$) can only be created by blending two (imprecise) observed prototype vectors V_i^1 and V_j^2. We call this solution sparse for the following reasons. Firstly, taking the situation given in Figure 8.11 as an example, for the vector $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ that optimizes $\mathcal{M}_4(\mathbf{y}, \mathrm{conv}(C_1), \mathrm{conv}(C_2))$ we can write

$$\mathbf{c}_2^* = \beta_1 \, \mathbf{c}_2^{1*} + \beta_2 \, \mathbf{c}_2^{2*} + \beta_3 \, \mathbf{c}_2^{3*} \tag{8.46}$$

where $\mathbf{c}_2^{1*} \in V_1^2$, $\mathbf{c}_2^{2*} \in V_2^2$, $\mathbf{c}_2^{3*} \in V_3^2$, $\beta_1 + \beta_2 + \beta_3 = 1$ and $\beta_i \geq 0$. Moreover, as can be seen from this figure, both β_1 and β_2 are strictly positive. As such, we call this solution non-sparse. Secondly, for the vector $(x_1^\bullet, \mathbf{c}_1^\bullet, \mathbf{c}_2^\bullet)$ that optimizes $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$ we have that

$$\mathbf{c}_2^\bullet = \alpha_1 \, \mathbf{c}_2^{1\bullet} + \alpha_2 \, \mathbf{c}_2^{2\bullet} + \alpha_3 \, \mathbf{c}_2^{3\bullet} \,,$$

where $\mathbf{c}_2^{1\bullet} \in V_1^2$, $\mathbf{c}_2^{2\bullet} \in V_2^2$, $\mathbf{c}_2^{3\bullet} \in V_3^2$, $\alpha_1 + \alpha_2 + \alpha_3 = 1$ and $\alpha_i \geq 0$. Moreover, in this case, we can choose $\mathbf{c}_2^{1\bullet}, \mathbf{c}_2^{2\bullet}$ and $\mathbf{c}_2^{3\bullet}$ such that only α_3 is strictly positive. In that sense, we will obtain sparse solutions. This contrasts the original setting, where $\mathbf{c}_1^*$ and $\mathbf{c}_2^*$ can both be blends of multiple (imprecise) observed prototype vectors. Indeed, in some applications this can be considered as unlikely. As such, the presented approach puts an additional emphasis on the data that is used to represent the classes.

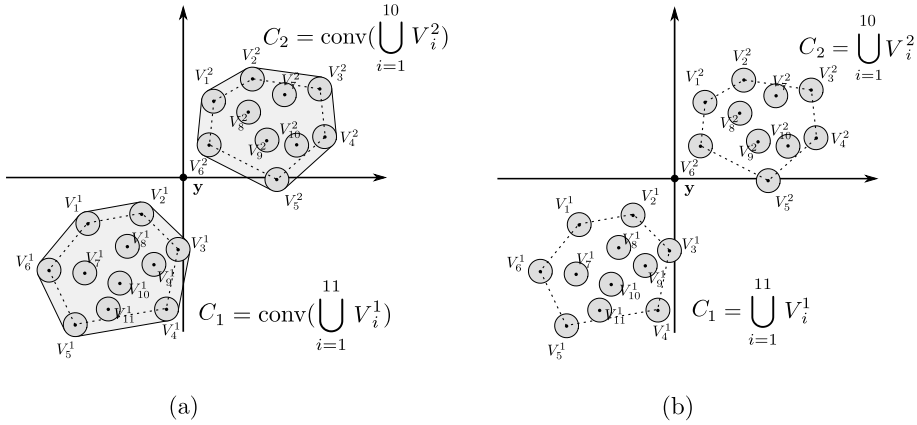


Figure 8.10: (a) C_1 and C_2 are defined as the convex hulls of ball-shaped sets. The dotted line gives the convex hulls of the centers $(\mathbf{v}_i^j)$ of the sets V_i^j. (b) C_1 and C_2 are defined as the union of ball-shaped sets.

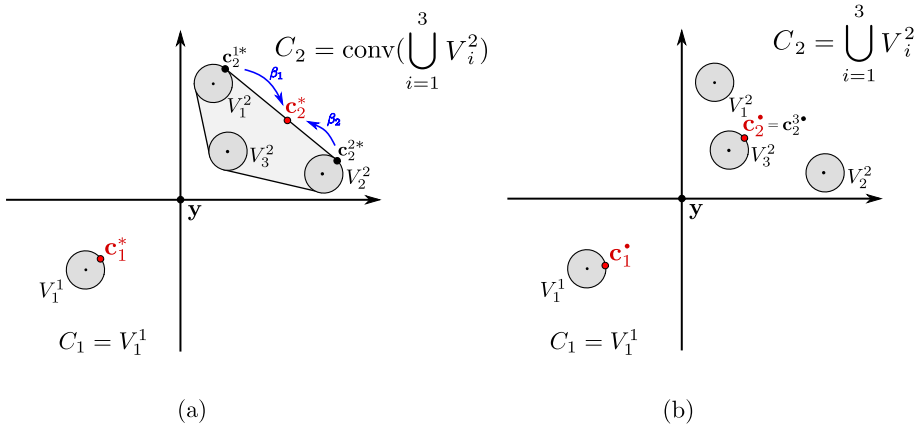


Figure 8.11: Illustration of the sparseness of solutions: (a) Non-sparse case, $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is the vector that optimizes $\mathcal{M}_4(\mathbf{y}, \mathrm{conv}(C_1), \mathrm{conv}(C_2))$. $\beta_1\, \mathbf{c}_1^{1*} + \beta_2\, \mathbf{c}_2^{2*} = \mathbf{c}_2^*$ and $\beta_3 = 0$. (b) Sparse case, $(x_1^\bullet, \mathbf{c}_1^\bullet, \mathbf{c}_2^\bullet)$ is the vector that optimizes $\mathcal{M}_4(\mathbf{y}, C_1, C_2)$.

8.7.2. Optimizing with a sparsity constraint

From (8.45), it can easily be seen that $C_1, \ldots, C_n$ are not convex. Because of that, $\mathcal{M}_1$ cannot be solved efficiently using the approaches that have been described up to this point. More fundamentally, in this case we do not have the guarantee that $X^k(\mathbf{y}, (C_i)_{i=1}^n)$ is an interval. However, this does not prevent us from attempting to find $\sup(X^k(\mathbf{y}, (C_i)_{i=1}^n))$ here. In the remainder of this section, we will derive a branch and bound (B & B) procedure that can be used to search for $\sup(X^k(\mathbf{y}, (C_i)_{i=1}^n))$.

A B & B procedure for the case $n = 2$

We will first discuss the case $n = 2$. For simplicity, we will assume that $\mathbf{y} \in \operatorname{conv}(C_1 \cup C_2)$. Note that this does not guarantee that $X^k(\mathbf{y}, (C_i)_{i=1}^n) \neq \emptyset$. We can now use the fact that C_1 and C_2 are the union of convex sets to obtain the following equality:

$$\sup(X^k(\mathbf{y}, (C_i)_{i=1}^2)) = \max_{\substack{i=1,\ldots,m_1 \\ j=1,\ldots,m_2}} h(\mathbf{y}, V_i^1, V_j^2)\,. \tag{8.47}$$

A brute-force approach would require $m_1 \times m_2$ function evaluations of h. When both m_1 and m_2 are large, this can become computationally quite demanding. Moreover, when $n > 2$, the complexity of this brute-force approach increases rapidly (when $m_1 = m_2 = \ldots = m_n$, the complexity is exponential in n). Because of that, we propose a branch and bound procedure for solving (8.47). Within this procedure, the sets C_1 and C_2 will be split recursively by a branching rule to obtain more restricted optimization problems. Hereafter, we describe the upper bound, lower bound, branching rule and pruning rule that are used within the B & B procedure.

Let $I_1 \subseteq \{1, \ldots, m_1\}$ and $I_2 \subseteq \{1, \ldots, m_2\}$ be two index sets. We can now denote

$$C_1^t = \bigcup_{i \in I_1} V_i^1\,, \qquad \text{and} \qquad C_2^t = \bigcup_{i \in I_2} V_i^2\,,$$

as the sets that are used at the t-th node of the B & B tree.

Upper bound: It can easily be seen that $h(\mathbf{y}, \operatorname{conv}(C_1^t), \operatorname{conv}(C_2^t))$ bounds $\sup(X^k(\mathbf{y}, C_1^t, C_2^t))$ from above. Indeed, as $C_1^t \subseteq \operatorname{conv}(C_1^t)$ and $C_2^t \subseteq \operatorname{conv}(C_2^t)$, $\mathcal{M}_4(\mathbf{y}, \operatorname{conv}(C_1^t), \operatorname{conv}(C_2^t))$ is a relaxation of $\mathcal{M}_4(\mathbf{y}, C_1^t, C_2^t)$. Moreover, if $h(\mathbf{y}, \operatorname{conv}(C_1^t), \operatorname{conv}(C_2^t)) = -\infty$, we can decide that this node is infeasible.

Lower bound: In general, any feasible point can be used as a lower bound on $\sup(X^k(\mathbf{y}, (C_i^t)_{i=1}^2))$. Unfortunately, the search for feasible points can be quite demanding in the general case. Instead, we propose the following strategy:

1. Firstly, let the optimum of $\mathcal{M}_4(\mathbf{y}, \text{conv}(C_1^t), \text{conv}(C_2^t))$ be given by $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$; if both

$$\mathbf{c}_1^* \in V_k^1 \text{ for some } k \in I_1\,, \qquad \text{and} \qquad \mathbf{c}_2^* \in V_\ell^2 \text{ for some } \ell \in I_2\,,$$

   it follows that the point $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is feasible for $\mathcal{M}_4(\mathbf{y}, C_1^t, C_2^t)$. Moreover, as $x_1^* = h(\mathbf{y}, \text{conv}(C_1^t), \text{conv}(C_2^t))$, it is globally optimal for $\mathcal{M}_4(\mathbf{y}, C_1^t, C_2^t)$. This means that the t-th node should not be split further.

2. Secondly, if $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ is not feasible, we could use heuristics (or an exhaustive search) to find a feasible point (or conclude that such a point does not exist). Such a feasible point could then be used to obtain an ϵ-bound on the optimality of a solution. However, due to the computational effort that is needed to accomplish this, we will not attempt to do so. Instead, the t-th node will remain active as a candidate for splitting. If at some point a feasible value is found for one of the descendants of the t-th node, the t-th node can inherit that lower bound.

Pruning rule: We will only be using a very simple pruning rule. Let the best feasible value for x_1 that has been found up to a given point be denoted x_1^{L}. Any (active) node t for which $h(\mathbf{y}, \text{conv}(C_1^t), \text{conv}(C_2^t)) < x_1^{\text{L}}$ is pruned.

Branching rule: In the branching phase (also see Figure 8.12) we will, from all active nodes, select the one with the highest upper bound. Let the t-th node be the one that is selected for branching. Moreover, let $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_2^*)$ be optimal for $\mathcal{M}_4(\mathbf{y}, \text{conv}(C_1^t), \text{conv}(C_2^t))$. As the t-th node is active, we know that

$$\mathbf{c}_1^* \notin \bigcup_{k \in I_1} V_k^1 \qquad \text{or} \qquad \mathbf{c}_2^* \notin \bigcup_{k \in I_2} V_k^2\,.$$

Without loss of generality, we can assume that $\mathbf{c}_2^* \notin \bigcup_{k \in I_2} V_k^2$.

We now choose a partitioning $\{I_2^{\text{L}}, I_2^{\text{R}}\}$ of the set I_2, and use this partitioning to define the following sets:

$$C_2^{\text{L}_t} = \bigcup_{i \in I_2^{\text{L}}} V_i^2\,, \qquad \text{and} \qquad C_2^{\text{R}_t} = \bigcup_{i \in I_2^{\text{R}}} V_i^2\,.$$

The sets $\{C_1^t, C_2^{\text{L}_t}\}$ and $\{C_1^t, C_2^{\text{R}_t}\}$ are then used to construct the two descendants of the t-th node. The partitioning $\{I_2^{\text{L}}, I_2^{\text{R}}\}$ can be constructed in several manners. Naturally, would like to select a partitioning rule that leads to a B & B procedure that will converge quickly. To achieve this, we will take the following rules into account:

1. It should hold that $\mathbf{c}_2^* \notin \text{conv}(C_2^{\text{L}_t})$ and $\mathbf{c}_2^* \notin \text{conv}(C_2^{\text{R}_t})$

2. The size of both $\text{conv}(C_2^{\text{L}_t})$ and $\text{conv}(C_2^{\text{R}_t})$ should be small.

3. Sets $\mathrm{conv}(C_2^{\mathrm{L}_t})$ and $\mathrm{conv}(C_2^{\mathrm{R}_t})$ that are "lengthy" in the direction of the line segment $\overline{(\mathbf{y}, \mathbf{c}_2^*)}$ are preferred over sets that are lengthy in a direction orthogonal to $\overline{(\mathbf{y}, \mathbf{c}_2^*)}$.
4. We should have $|I_2^{\mathrm{L}}| \approx |I_2^{\mathrm{R}}|$.

These rules were constructed intuitively, to minimize the number of splits needed in the B & B procedure. We now propose a simple heuristic that provides a partitioning, taking the rules above into account. Firstly, we reduce the ball-shaped sets V_k^2 with $(k \in I_2)$ to their centers $\mathbf{v}_k^2$. Secondly, we define a hyperplane (in the q-dimensional space) passing through[5] $\mathbf{c}_2^*$ and $\mathbf{y}$. This hyperplane can be used to separate the vectors $\mathbf{v}_k^2$. Let $\mathbf{z}$ be a normal vector of this hyperplane. We now choose the sets I_2^{L} and I_2^{R} as follows:

$$I_2^{\mathrm{L}} = \{i \in I_2 \mid \mathbf{z}^\top (\mathbf{v}_i^2 - \mathbf{y}) \leq 0\} \quad \text{and} \quad I_2^{\mathrm{R}} = \{i \in I_2 \mid \mathbf{z}^\top (\mathbf{v}_i^2 - \mathbf{y}) > 0\}\,.$$

A B & B procedure for the case $n > 2$

We now generalize the procedure above to the case $n > 2$. Most concepts of the case where $n = 2$ can be transferred. In this section, we elaborate on the most important differences and pitfalls.

Up to this point, disregarding the value for n, the computation of $\sup(X^k)$ always relied on a reformulation of the original optimization problem to an equivalent optimization problem that has only two classes. Unfortunately, the definition of the sets C_i in (8.45) severely complicates such a reformulation in the general case. However, as we show next, we can easily provide a relaxation of the original problem that does allow a reformulation. Without loss of generality, we assume that we want to compute $\sup(X^1)$. Moreover, assume that, at the t-th node in the B & B tree, we have the index sets $I_1, \ldots, I_n$, where $I_\ell \subseteq \{1, \ldots, m_\ell\}$. We now define C_1^t and $C_\bullet^t$ as:

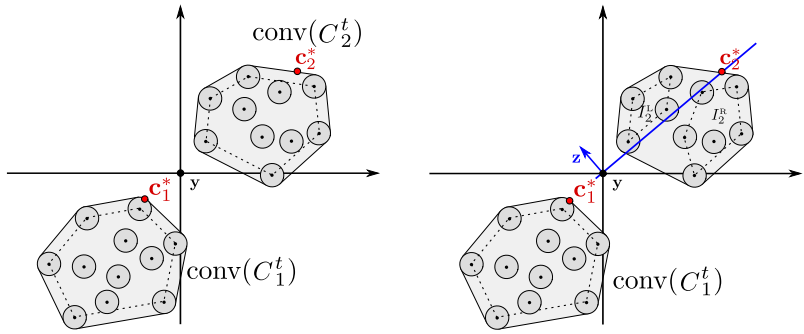
$$C_1^t = \bigcup_{i \in I_1} V_i^1\,, \qquad \text{for } k = 1 \ldots, n\,,$$

and

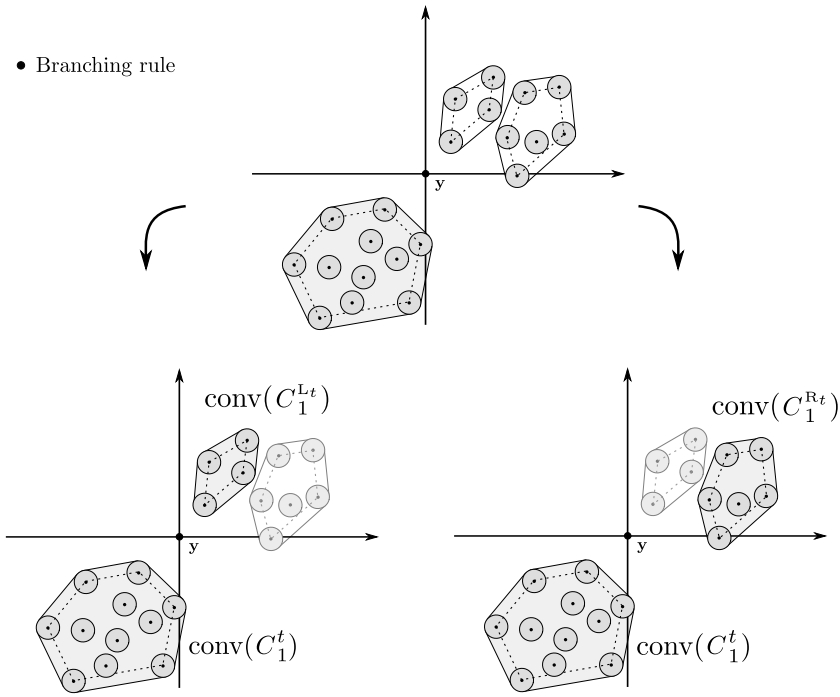
$$C_\bullet^t = \bigcup_{j=2}^{n} \Big(\bigcup_{i \in I_j} V_i^j \Big)\,.$$

[5] When $q > 2$, the hyperplane passing through $\mathbf{c}_2^*$ and $\mathbf{y}$ is not unique. In those cases, to define a unique hyperplane $q - 2$ additional points are needed. The first point that is included is the arithmetic mean of the prototype vectors belonging to the current node. When $q > 3$, the remaining points are randomly drawn from $[0, 1]^q$.

- Computation of a relaxed solution (upper bound)



- Branching rule



First descendant

Second descendant

Figure 8.12: The branching rule of the B & B procedure when $n = 2$.

Upper bound: Similar to the previous section, $h(\mathbf{y}, \mathrm{conv}(C_1^t), \mathrm{conv}(C_\bullet^t))$ can be used as an upper bound on $\sup(X^1(\mathbf{y}, (C_i)_{i=1}^n))$.

Lower bound: Let $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_\bullet^*)$ be the vector that optimizes $\mathcal{M}_4(\mathbf{y}, \mathrm{conv}(C_1^t), \mathrm{conv}(C_\bullet^t))$. We now have two options:

1. When $\mathbf{c}_1^* \notin C_1^t$: the relaxed solution is not feasible for the original problem. In this case, the second step is not considered. The node remains active.

2. When $\mathbf{c}_1^* \in C_1^t$: We can always find a set of k non-identical vectors $\{\mathbf{z}_1, \ldots, \mathbf{z}_k\} \subset C_\bullet^t$ (where $k \leq q$) that can be used to decompose $\mathbf{c}_\bullet^*$ as follows

$$\mathbf{c}_\bullet^* = \sum_{i=1}^{k} \gamma_i \, \mathbf{z}_i \, ,$$

   such that $\sum_{i=1}^k \gamma_i = 1$, $\gamma_i > 0$. As $\mathbf{c}_\bullet^*$ is located at the boundary of $\mathrm{conv}(C_\bullet^t)$, this decomposition is unique. Moreover, $\{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$ can easily be obtained from the results of the optimization procedure that computes the upper bound. If, for each of the $n-1$ classes that contribute to $C_\bullet^t$, we have at most one $\mathbf{z}_i \in C_k^t$, then $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_\bullet^*)$ represents a solution that is feasible for the original optimization problem. In that case, we do not need to branch the current node any further. Moreover, $(x_1^*, \mathbf{c}_1^*, \mathbf{c}_\bullet^*)$ is optimal for that branch.

Pruning rule: This rule is identical to the case $n = 2$.

Branching rule: In the branching phase we will, from all active nodes, select the one with the highest upper bound. Here as well two options exist:

1. When $\mathbf{c}_1^* \notin C_1^t$: we partition the set C_1^t, using the same criteria as before. Let $\{I_1^{\mathrm{L}}, I_1^{\mathrm{R}}\}$ be a partitioning of I_1. This partitioning can be used to define the following sets:

$$C_1^{\mathrm{L}_t} = \bigcup_{i \in I_1^{\mathrm{L}}} V_i^1 \, , \qquad \text{and} \qquad C_1^{\mathrm{R}_t} = \bigcup_{i \in I_1^{\mathrm{R}}} V_i^1 \, .$$

   We can now use the sets $\{C_1^{\mathrm{L}_t}, C_2^t, \ldots, C_n^t\}$ and $\{C_1^{\mathrm{R}_t}, C_2^t, \ldots, C_n^t\}$ to construct the descendants of the t-th node.

2. When $\mathbf{c}_1^* \in C_1^t$ and the relaxed solution is not feasible: using the decomposition of $\mathbf{c}_\bullet^t$ above, we have at least one triplet of indices i, j, k such that both $\mathbf{z}_i, \mathbf{z}_j \in C_k^t$. We can now split C_k^t, using a procedure that is similar to the case $n = 2$. A hyperplane is constructed that can be used to separate $\mathbf{v}_\ell^k$. This hyperplane passes through the points $\mathbf{y}$, $\gamma_i \, \mathbf{z}_i + \gamma_j \, \mathbf{z}_j$ and $q-2$ additional points (see footnote). Note that $\gamma_i \, \mathbf{z}_i + \gamma_j \, \mathbf{z}_j$ is the equivalent of $\mathbf{c}_2^*$ in the case $q = 2$.

   Let $\{I_k^{\mathrm{L}}, I_k^{\mathrm{R}}\}$ be a resulting partitioning of I_k. This partitioning can be used

to define the following sets:

$$C_k^{\mathrm{L}_t} = \bigcup_{i \in I_k^{\mathrm{L}}} V_i^k \,, \qquad \text{and} \qquad C_k^{\mathrm{R}_t} = \bigcup_{i \in I_k^{\mathrm{R}}} V_i^k \,.$$

We can now use the sets $\{C_1^t, \ldots, C_k^{\mathrm{L}_t}, \ldots, C_n^t\}$ and $\{C_1^t, \ldots, C_k^{\mathrm{R}_t}, \ldots, C_n^t\}$ to construct the descendants of the t-th node.

The B & B procedure described above has a worst-case complexity that is equal to the complexity of the exhaustive procedure. However, in practice, the worst case scenario seldom occurs.

8.8. Conclusions and discussion

In this chapter, a series of optimization problems were proposed that can be used to compute the interval estimator that was introduced in the previous chapter in an efficient manner (Objectives III.2 and III.3). Moreover, the interval estimating procedure is extended to allow settings in which sources have a high-dimensional representation (Objective III.4). Finally, an estimation procedure that forces sparsity in the solution was proposed (Objective III.4).

The general methodology and the optimization problems that were proposed in this chapter allow interval estimators for the unmixing of imprecisely described sources to be computed in a variety of settings. We mainly focused on the development of a flexible methodology that provides a sound way to work with uncertainty when estimating the proportional contribution of a source to a mixture. Naturally, such an estimate can only be accurate when the set-based description of the sources is sufficiently precise. Several approaches were proposed that can be used to derive a set-based description from data. However, the theoretical basis for these approaches is rather limited. For example, we argued why the convex hull of a set of datapoints is probably a suboptimal estimate of the "true" set that describes a source. Several alternatives were presented. Nevertheless, it remains an open question how these datapoints can be optimally translated into a set.

When the set-based description of a source is derived from a dataset (by taking the convex hull, or one of the alternatives that were proposed) the influence of a particular (extreme) point in that dataset may heavily influence the interval estimate that is obtained. This can be considered as a potential threat for our methodology especially in situations where outliers are likely to occur. In those cases, it may be interesting to attribute a grade of belief to the intervals that are obtained. Intervals that are supported by a large part of the dataset (*i.e.* the removal of a few points does not influence the interval estimate too heavily) could receive a higher grade of belief than intervals that are only supported by a limited number of extreme datapoints. Similarly, in several applied settings, the

datapoints may be given a grade of belief (for example, datapoints obtained through a thorough chemical analysis of a sample may be graded higher than datapoints that are found in some on-line data repository). This differential grading may be taken into account when attributing a degree of belief to an interval.

In this chapter, we limited our discussion to linear mixing models. However, as illustrated in the previous chapter, several processes require nonlinear mixing models. A direct extension of the estimators that were proposed here will probably lead to optimization problems that are hard to solve (even finding a feasible point may prove to be difficult). The development of a procedure that generalizes the current methodology towards those settings may prove challenging.

8.A. Equivalent linear programs and SOCPs

In Section 8.4.5 it was argued that, in some cases, $\mathcal{M}_8$ can be transformed into an equivalent linear program or second order cone program. In this appendix, this transformation is briefly described. The transformation that is used here is generally known as the Charnes and Cooper transformation [113]. It was originally developed for transforming a linear fractional program into a linear program. When the sets C_1 and C_2 are convex polytopes, $\mathcal{M}_8$ reduces to a linear fractional program and the Charnes and Cooper transformation can be applied directly.

Instead of presenting the complete transformation, we apply it directly to $\mathcal{M}_8$. As a starting point, consider the following tranformation of the vector of optimization variables $(b_1, b_2, \mathbf{a}^\top)^\top$:

$$\mathbf{z} = \frac{1}{b_2 - b_1} \begin{pmatrix} b_1 \\ b_2 \\ \mathbf{a} \end{pmatrix} .$$

It can easily be seen that, using this variable, the objective function of $\mathcal{M}_8$ can be rewritten as

$$\begin{pmatrix} 0 & 1 & -\mathbf{y}^\top \end{pmatrix} \mathbf{z} .$$

The inequality constraints can be rewritten as:

$$\begin{aligned}
\begin{pmatrix} -1 & 0 & \mathbf{c}_1^\top \end{pmatrix} \mathbf{z} &\leq 0 , \quad \text{for all}\, \mathbf{c}_1 \in C_1 , \\
\begin{pmatrix} 0 & -1 & \mathbf{c}_2^\top \end{pmatrix} \mathbf{z} &\leq 0 , \quad \text{for all}\, \mathbf{c}_2 \in C_2 , \\
\begin{pmatrix} 1 & 0 & -\mathbf{y}^\top \end{pmatrix} \mathbf{z} &< 0 , \\
\begin{pmatrix} 0 & -1 & \mathbf{y}^\top \end{pmatrix} \mathbf{z} &< 0 .
\end{aligned}$$

However, $\mathbf{z}$ is not a free variable. The transformation that is used in the definition of $\mathbf{z}$ implies that

$$\begin{pmatrix} -1 & 1 & \mathbf{0}_q^\top \end{pmatrix} \mathbf{z} = \begin{pmatrix} -1 & 1 & \mathbf{0}_q^\top \end{pmatrix} \frac{1}{b_2 - b_1} \begin{pmatrix} b_1 \\ b_2 \\ \mathbf{a} \end{pmatrix} = 1 .$$

Therefore, the constraint

$$\begin{pmatrix} -1 & 1 & \mathbf{0}_q^\top \end{pmatrix} \mathbf{z} = 1 ,$$

should be included as well.

This transformation leads to the following optimization problem:

$$
\begin{aligned}
\mathcal{M}_{11}: \quad & \underset{\mathbf{z}\in\mathbb{R}^{q+2}}{\text{minimize}} && \begin{pmatrix} 0 & 1 & -\mathbf{y}^\top \end{pmatrix} \mathbf{z} \\
& \text{subject to} && \begin{pmatrix} -1 & 0 & \mathbf{c}_1^\top \\ 0 & -1 & \mathbf{c}_2^\top \end{pmatrix} \mathbf{z} \quad \leq \mathbf{0}_2\,, \text{ for all } \mathbf{c}_1 \in C_1 \text{ and } \mathbf{c}_2 \in C_2\,, \\
& && \begin{pmatrix} 1 & 0 & -\mathbf{y}^\top \\ 0 & -1 & \mathbf{y}^\top \end{pmatrix} \mathbf{z} \quad \leq \mathbf{0}_2\,, \\
& && \begin{pmatrix} -1 & 1 & \mathbf{0}_q^\top \end{pmatrix} \mathbf{z} \quad = 1\,,
\end{aligned}
$$

The equivalence between $\mathcal{M}_8$ and $\mathcal{M}_{11}$ follows from the general result of [113].

Unfortunately, as $\mathcal{M}_8$, the new optimization problem $\mathcal{M}_{11}$ has an infinite number of inequality constraints. However, as argued before, when C_1 is a convex polytope,

$$\begin{pmatrix} -1 & 0 & \mathbf{c}_1^\top \end{pmatrix} \mathbf{z} \leq 0\,, \quad \text{for all}\, \mathbf{c}_1 \in C_1\,,$$

reduces to

$$\begin{pmatrix} -1 & 0 & \mathbf{c}_1^\top \end{pmatrix} \mathbf{z} \leq 0\,, \quad \text{for each vertex}\, \mathbf{c}_1 \text{ of } C_1\,.$$

Moreover, when C_1 is ellipsoidal, *i.e.*

$$C_1 = \{\mathbf{v}_1 + \mathbf{V}^1\mathbf{u} \mid \mathbf{u} \in \mathbb{R}^q \wedge \|\mathbf{u}\|_2 \leq 1\}\,,$$

where $\mathbf{v}_1$ is a q-vector and $\mathbf{V}^1$ a $q \times q$ positive definite matrix,

$$\begin{pmatrix} -1 & 0 & \mathbf{c}_1^\top \end{pmatrix} \mathbf{z} \leq 0\,, \quad \text{for all}\, \mathbf{c}_1 \in C_1\,,$$

reduces to

$$\mathbf{v}_1^\top \begin{pmatrix} z_3 \\ \vdots \\ z_{q+2} \end{pmatrix} + \left\| \mathbf{V}^1 \begin{pmatrix} z_3 \\ \vdots \\ z_{q+2} \end{pmatrix} \right\|_2 \leq z_1\,.$$

This inequality is a second order cone constraint.

9 Set-based unmixing of mixtures in practice

9.1. Introduction

In Chapter 7, a set estimator for the proportional contribution of a source to a mixture was introduced. Moreover, it was argued that this estimator is somehow complementary to the more traditional point estimators and probability distribution estimators. In Chapter 8, several mathematical optimization problems were developed that allow an efficient computation of the set estimator. Moreover, multiple methodological extensions were made that resulted in robust or sparse versions of the set estimator. In the current chapter, we experiment with the methodology that was described in Chapters 7 and 8. More precisely, this chapter is organized as follows:

- In Section 9.2, the methodology described in Chapters 7 and 8 is illustrated by means of series of experiments on synthetic data. The set estimator is compared with point estimators and probability distribution estimators. Moreover, the influence of robustness and sparseness on the estimator is illustrated.
- In Section 9.3, the set estimator is applied to detect fraudulent levels of adulteration in vegetable oils.
- In Section 9.4, conclusions are presented and discussed.

9.2. Experiments on synthetic data

9.2.1. Point estimates, probability distribution estimates and set estimates

As a starting point of this section, consider the convex polyhedral sets $C_1, C_2 \subset \mathbb{R}^2$ in Figure 9.1 and Table 9.1 that represent two (imprecisely described) sources (referred to as source 1 and source 2). These sources have been used to create a mixture with (observed) mixture vector $\mathbf{y} = (0.4, 0.4)^\top$.

Set estimator

As the feasible sets C_1 and C_2 are convex polyhedrons of which the vertices are known, $X^1(\mathbf{y}, C_1, C_2)$ can be computed using the following optimization problem

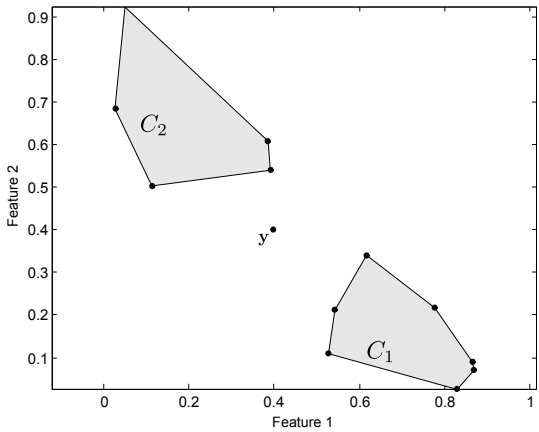


Figure 9.1: Visualization of the experiment with synthetic data. The coordinates of the vertices of the polyhedrons are given in Table 9.1.

Table 9.1: Synthetic data representing the observed prototypes of two sources.

Source number	Feature 1	Feature 2	Source number	Feature 1	Feature 2
1	0.5264	0.1082	1	0.5435	0.2047
1	0.8298	0.0266	2	0.1125	0.5038
1	0.8677	0.0708	2	0.3911	0.5399
1	0.8648	0.0800	2	0.3849	0.6085
1	0.7780	0.2179	2	0.0502	0.9242
1	0.6153	0.3398	2	0.0268	0.6835

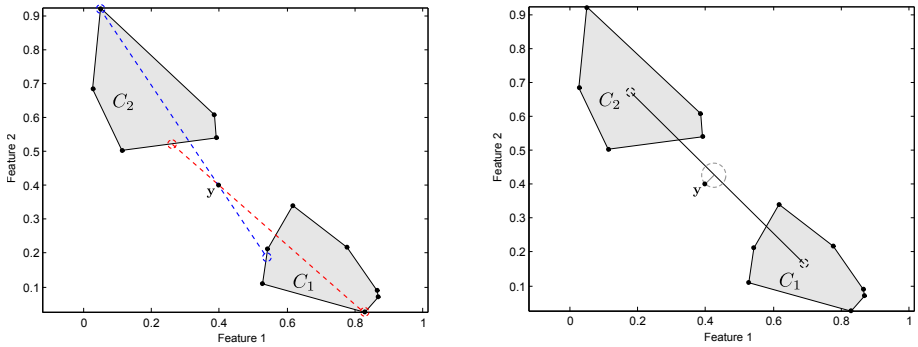


Figure 9.2: (a) Visualization of the location of the 'optimal' prototype vectors corresponding to the minimal (red) and maximal (blue) elements of X^1. (b) Visualization of the least squares point estimator.

(cfr. $\mathcal{M}_8$)

$$
\begin{aligned}
\underset{\mathbf{a}\in\mathbb{R}_0^d, b_1, b_2\in\mathbb{R}}{\text{minimize}} \quad & \frac{b_2 - \langle \mathbf{a}, \mathbf{y}\rangle}{b_2 - b_1} \\
\text{subject to} \quad & b_1 \geq \langle \mathbf{c}_1, \mathbf{a}\rangle \,, \qquad \text{for all vertices } \mathbf{c}_1 \in C_1 \,, \\
& b_2 \geq \langle \mathbf{c}_2, \mathbf{a}\rangle \,, \qquad \text{for all vertices } \mathbf{c}_2 \in C_2 \,, \\
& b_1 < \langle \mathbf{a}, \mathbf{y}\rangle \,, \\
& b_2 > \langle \mathbf{a}, \mathbf{y}\rangle \,.
\end{aligned}
$$

As this programme is a linear fractional program, it can be transformed into an equivalent linear program (see Appendix 8.A). The resulting linear program can be solved using any linear programming solver. Here, we used the `linprog` routine in Matlab, resulting in $X^1(\mathbf{y}, C_1, C_2) \approx [0.247, 0.713]$ (where we used $\approx$ to denote potential rounding errors). Figure 9.2(a) shows the location of the 'optimal' prototype vectors corresponding to the minimal (red) and maximal (blue) elements of X^1.

Point estimator (least squares)

Recall from Chapter 7 that the least squares estimator requires a precise description of the sources. To derive such a representation, we could for example use the center of mass of the polyhedrons C_1 and C_2. These are given by $\bar{\mathbf{c}}_1 \approx (0.69, 0.16)^\top$ and $\bar{\mathbf{c}}_2 \approx (0.18, 0.67)$. These points are shown in Figure 9.2(b). Subsequently, the following optimization problem is solved

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^2}{\text{minimize}} \quad & \left\| \mathbf{y} - \textstyle\sum_{i=1}^{2} x_i\, \bar{\mathbf{c}}_i \right\|_2^2 \\ \text{subject to} \quad & x_1 + x_2 = 1\,, \\ & x_1, x_2 \geq 0\,. \end{aligned}$$

The minimizer of this optimization problem is the point estimator $\mathbf{x}^{\text{LS}}$. This is a linearly constrained convex quadratic optimization problem that can be solved by any quadratic programming solver. We used the `quadprog` routine in Matlab, resulting in $\hat{x}_1 = 0.484$. In this example, the least squares estimate lies within the interval obtained by the set estimator. However, this is not the case in general.

We now provide a geometrical interpretation of the least squares estimator (see also Figure 9.2(b)). As $\mathbf{y}$ does not lie on the line segment that connects $\bar{\mathbf{c}}_1$ and $\bar{\mathbf{c}}_2$ (denoted $\overline{(\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)}$), $\mathbf{y}$ cannot be written as a convex combination of these centers of mass. Therefore, the least squares estimator locates the vector $\mathbf{y}' \in \overline{(\mathbf{c}_1, \mathbf{c}_2)}$ that minimizes the Euclidean distance between $\mathbf{y}'$ and $\mathbf{y}$. Once $\mathbf{y}'$ has been found, the proportional contribution of the first source to $\mathbf{y}'$ is

$$\frac{\|\bar{\mathbf{c}}_2 - \mathbf{y}\|_2}{\|\bar{\mathbf{c}}_1 - \mathbf{y}\|_2 + \|\bar{\mathbf{c}}_2 - \mathbf{y}\|_2}\,.$$

This value is equal to x_1^{LS}. Therefore, the least squares estimator implicitly assumes that $\mathbf{y}$ is a noisy observation of the true vector $\mathbf{y}'$. Moreover, the distribution of the noise vector $\mathbf{y} - \mathbf{y}'$ is assumed to be isotropic.

Probability distribution estimator

To derive the probability distribution estimator of x_1, each source is described by means of a bivariate random vector. Let these bivariate random vectors be $\mathcal{C}^1 = (\mathcal{C}_1^1, \mathcal{C}_2^1)$ for the first source and $\mathcal{C}^2 = (\mathcal{C}_1^2, \mathcal{C}_2^2)$ for the second source. Moreover, the mixing proportions are random variables as well. Let $\mathcal{X} = (\mathcal{X}_1, \mathcal{X}_2)$ be the random vector of mixing proportions. A mixture vector is obtained using the following scheme:

1. Draw an observation $\mathbf{c}_1$ from $\mathcal{C}^1$ (the first prototype vector).
2. Draw an observation $\mathbf{c}_2$ from $\mathcal{C}^2$ (the second prototype vector).
3. Draw an observation (x_1, x_2) from $\mathcal{X}$ (the mixing proportions).
4. Define $\mathbf{y}$ using the LMM: $\mathbf{y} = x_1\mathbf{c}_1 + x_2\mathbf{c}_2$.

According to this generative scheme, the mixing vector itself is modeled through a bivariate random vector $\mathcal{Y}$ that is defined as follows: $\mathcal{Y} = \mathcal{X}_1\,\mathcal{C}^1 + \mathcal{X}_2\,\mathcal{C}^2$. The probability distribution estimator is the conditional probability density function $f_{\mathcal{X}_1|\mathcal{Y}}$.

Finding a closed-form solution for $f_{\mathcal{X}_1|\mathcal{Y}}$ is often a difficult task. Therefore, we resort to stochastic simulation to obtain an approximation of $f_{\mathcal{X}_1|\mathcal{Y}}$. We proceed in the following manner.

1. Simulate a large sample of the random vector $(\mathcal{X}_1, \mathcal{Y})$.
2. Use a kernel density estimator to obtain an approximation $\hat{f}_{\mathcal{X}_1,\mathcal{Y}}$ (resp. $\hat{f}_{\mathcal{Y}}$) of $f_{\mathcal{X}_1,\mathcal{Y}}$ (resp. $f_{\mathcal{Y}}$)
3. Use $\hat{f}_{\mathcal{X}_1,\mathcal{Y}}/\hat{f}_{\mathcal{Y}}$ to approximate $f_{\mathcal{X}_1|\mathcal{Y}}$.

For $\mathcal{C}^1$ (resp. $\mathcal{C}^2$) uniformly distributed over C_1 (resp. C_2), $\mathcal{X}_1$ uniformly distributed over $[0,1]$ (with $\mathcal{X}_2 = 1 - \mathcal{X}_1$), and assuming that $\mathcal{C}^1$, $\mathcal{C}^2$ and $\mathcal{X}$ are mutually independent random vectors, Figure 9.3 (solid line) shows an approximation[1] of $f_{\mathcal{X}_1|\mathcal{Y}}(. \mid (0.4, 0.4))$. From this figure, it can be seen that the support of the approximated conditional probability density function approximates the interval obtained by the set estimator (the fact that the support is somewhat larger is due to approximation errors of the density estimation). Moreover, in addition to the support, the distribution estimator gives information about the likelihood of the mixing proportions. Therefore, it could be argued that the distribution estimator is more informative than the set estimator. However, the increased information content comes at some cost. Firstly, the added information is only (approximately) correct if the assumption of uniformly distributed source vectors (approximately) holds. Secondly, we are only able to obtain an approximation of the conditional probability density function. Unfortunately, the approximation methods scale badly. An increase of the dimensionality of the source vectors quickly leads to large approximation errors and excessive computational requirements.

To illustrate the influence of the distribution of the sources, the result of a second experiment is given (dotted line in Figure 9.3). In this experiment, $\mathcal{C}^1$ and $\mathcal{C}^2$ are bivariate normally distributed. The mean vectors and the covariance matrices of these distributions were the sample means and the sample covariances of the vertices of the polytopes C_1 and C_2. Comparing the resulting estimate with the estimate obtained for the uniform distributions, it can clearly be seen that the conditional distribution function is heavily influenced by the (assumed) distribution of the sources.

9.2.2. The influence of robustness

In the previous section, the sets C_1 and C_2 were defined as the convex hulls of two collections of (observed) prototype vectors. In Section 8.5 of the previous chapter, it was argued that, in some cases, this approach can be too restrictive.

[1] To obtain this approximation, a sample with sample size 1000 was simulated using a rejection sampling algorithm. For the kernel density estimation, a Gaussian kernel was used. The bandwith of the kernel was computed using the `Hpi` routine in the R-package `ks` (version number 1.8.13 [114])

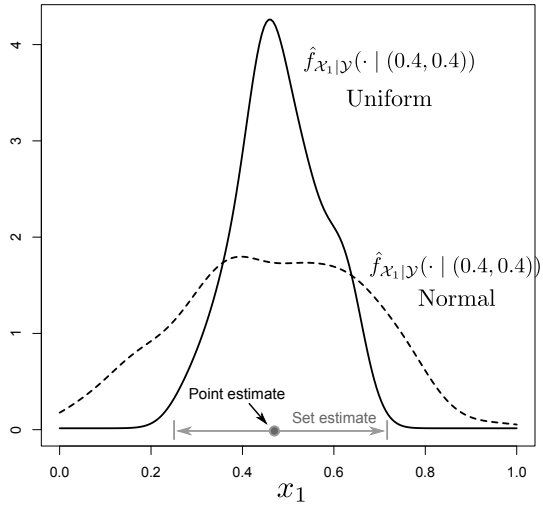


Figure 9.3: An approximation of the conditional probability density function $f_{\mathcal{X}_1|\mathcal{Y}}(. \mid (0.4, 0.4))$ for $\mathcal{C}_1$ (resp. $\mathcal{C}_2$) uniformly distributed over C_1 (resp. C_2) (solid line), and for $\mathcal{C}_1$ and $\mathcal{C}_2$ bivariate normally distributed (dotted line).

For example, when the observed prototype vectors are noisy, the obtained interval estimate can be inaccurate. Indeed, when the convex hulls of these noisy points are used as estimates (denoted $\hat{C}_1$ and $\hat{C}_2$) for C_1 and C_2, we generally have that $X^k(\mathbf{y}, C_1, C_2) \neq X^k(\mathbf{y}, \hat{C}_1, \hat{C}_2)$. As a result, the true mixing proportions may lie outside the estimated interval[2].

To overcome this problem, an alternative method for estimating C_1 and C_2 was proposed. In this setting, for each observed prototype vector, an ellipsoid is defined that encloses the observed vector. Subsequently, it is assumed that any point within the ellipsoid is a valid observed prototype vector. As this procedure is applied to each prototype vector, two sets of ellipsoids are obtained. Consequently, C_1 and C_2 are estimated by the convex hulls of these sets (for a formal description of this procedure, we refer to Section 8.5).

We now apply this procedure to the synthetic problem from the previous section (see Figure 9.4). Unfortunately, the synthetic problem does not provide a natural way to define the ellipsoids. Therefore, we decided (somewhat arbitrary) to use the eigenvectors of the covariance matrix of the observed prototype vectors[3] as directions for the main axes of the ellipsoids. Moreover, the ellipsoids were centered at the observed prototype vectors and the lengths of the axes were chosen proportional to

[2] Alternatively, the obtained interval may be too wide as well. Unfortunately, as argued in Section 8.5, this case does not lead to an optimization problem that can be solved efficiently. Therefore, we do not consider this case in depth.

[3] The covariance matrix of the prototypes of the first and the second class were computed independently. As a result, the orientation of the ellipsoids of the two classes is different.

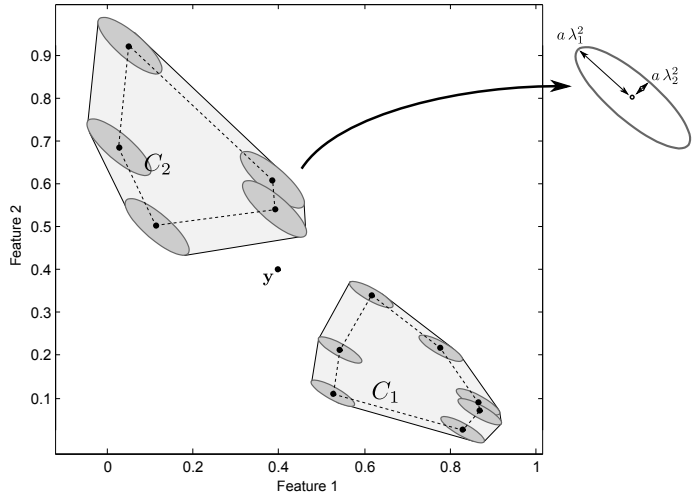


Figure 9.4: Enclosing of the prototype vectors in ellipsoids. The sets C_1 and C_2 are estimated by means of the convex hulls of these ellipsoids (shaded areas). The lengths of the axes of the ellipsoids are proportional to their eigenvalues (the eigenvalues are denoted $\lambda_1^1 \approx 0.028$, $\lambda_2^1 = 0.007$, $\lambda_1^2 \approx 0.046$ and $\lambda_2^2 = 0.014$). The axes lengths are $a\,\lambda_1$ and $a\,\lambda_2$, where a is a positive constant.

the eigenvalues (λ_1^1 and λ_2^1 for the first class and λ_1^2 and λ_2^2 for the second class) of the covariance matrix. More precisely, the lengths of the axes are $a\,\lambda_1^1$ and $a\,\lambda_2^1$ where $a \in \mathbb{R}^+$ (resp. $a\,\lambda_1^2$ and $a\,\lambda_2^2$).

To illustrate the influence of this procedure on the interval estimate, Figure 9.5 shows the interval estimate for a ranging in the interval $[0, 10]$. Naturally, $\inf(X^1)$ is a decreasing function of a and $\sup(X^1)$ is an increasing function of a. Moreover, this figure suggests a linear relationship between a and $\inf(X^1)$ and $\sup(X^1)$. However, we have no theoretical guarantees regarding the linearity of this relationship.

9.2.3. Sparse estimates

The discussion in this section describes some of the principles that are introduced in Section 8.7 of the previous chapter. The following description can be seen as a wordy version of Section 8.7.

Up to this point, the observed prototype vectors are used only to define the convex hulls needed to estimate C_1 and C_2. Essentially, this approach uses the information that is present in the data to define a set. It could be argued that this approach has at least two downsides. Firstly, a considerable part of the information present in the data is ignored by the set estimator. Secondly, even though the obtained interval is both mathematically and semantically correct, it may imply that mixtures are generated using mixing procedures that are unlikely to occur in practice. To make

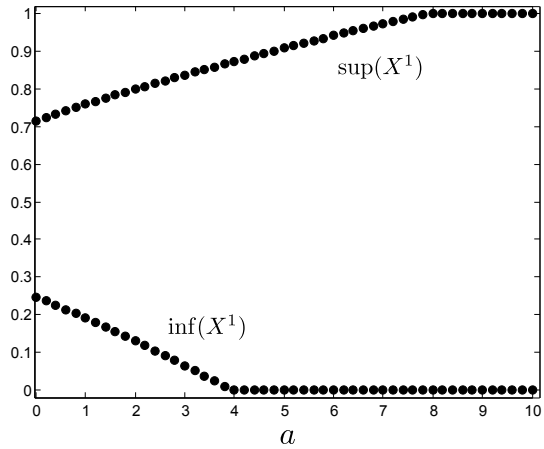


Figure 9.5: Evolution of $\inf(X^1)$ and $\sup(X^1)$ as a function of a, for the experimental setup described in Figure 9.4.

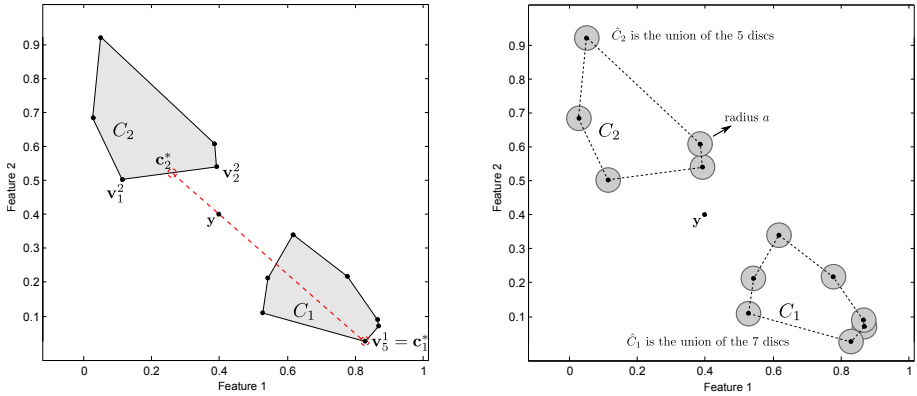
this more clear, consider the situation depicted in Figure 9.6(a). Essentially, this figure is a copy of Figure 9.2(a), with some additional formatting. To obtain $\sup(X^1) \approx 0.713$, it is assumed that the mixture with mixture vector $\mathbf{y}$ *can* be obtained as follows:

1. Use the observed prototype vectors $\mathbf{v}_1^2$ and $\mathbf{v}_2^2$ to create a mixture with mixture vector $\mathbf{c}_2^*$.
2. Create the mixture with mixture vector $\mathbf{y}$ by mixing $\mathbf{c}_2^*$ and $\mathbf{v}_5^1$:

$$\mathbf{y} = 0.713\,\mathbf{v}_5^1 + 0.287\,\mathbf{c}_2^*\,.$$

The above procedure implies that, at some point, an 'intermediate' mixture ($\mathbf{c}_2^*$) is created. Depending on the application at hand this may be implausible. For example, consider an application where two types of vegetable oil are blended (we consider this example in detail in the following section). It could seem nonrealistic that a producer of vegetable oil will blend two pure oils of the same type prior to blending those oils with a third vegetable oil of another type. In this small example, that strategy may still be reasonable. However, in general the intermediate mixture will often be a blend of q (where q is the dimensionality of the problem) observed mixtures. As the dimensionality (q) of the problem increases, the number of prototypes that is needed may become nonrealistically high.

To overcome these problems, we can constrain the number of observed prototypes that is used to create the intermediate mixture. However, to be able to do this, we are forced to assume that the observed prototype vectors are noisy observations of the true prototype vectors (using the principles from the previous subsection). As an example, consider the case where the intermediate mixture is constrained to



(a) (b)

Figure 9.6: (a) Visualization of the geometry of a set estimator that uses the convex hulls of the observed prototypes. This visualization illustrates that, to construct a mixture with mixture vector $\mathbf{y}$ and a proportional contribution of the first source that is equal to $\inf(X^1)$, the source vectors $\mathbf{c}_2^*$ and $\mathbf{c}_1^* = \mathbf{v}_5^1$ are used. (b) Enclosing of the observed prototype vectors in discs with radius a. The sets C_1 and C_2 are estimated by as union of these discs (shaded areas).

be near one of the observed mixtures. This situation can be encoded by enclosing each observed prototype vector into a disc with radius a. Subsequently, the sets C_1 and C_2 are estimated as the union of these discs (see Figure 9.6). The estimator that is obtained in this manner is called a sparse estimator (for an explanation of this name, see Section 8.7).

To compute $\inf(X^k)$ and $\sup(X^k)$, the branch and bound procedure introduced in Section 8.7 can be used. Table 9.2 presents the results obtained for different radii a. From this table, we can see that as a goes to zero, the sparse estimate results in an empty set. However, to verify that this set is empty, only 13 second order cone programs (SOCPs) needed to be solved whereas a naive approach would require $5 \times 7 = 35$ SOCPs to be solved. Secondly, the difference between the sparse estimate and the robust estimate (using the convex hull) can be quite large. For example, for $a = 0.01$, the estimates for the infimum are rather different.

9.3. Demonstration: Detecting fraudulent levels of adulteration in vegetable oils

9.3.1. Problem setting and data

As already suggested in Chapter 7, the adulteration of (pure) vegetable oils such as for instance mustard oil with (often cheaper) other vegetable oils such as soybean

Table 9.2: Values obtained for $\inf(X^1)$ and $\sup(X^1)$ for the problem visualized in Figure 9.6 using different radii. Results are presented for the case where C_1 and C_2 are defined as the convex hulls of the discs in Figure 9.6(a) (2nd and 4th column) as well as the case where C_1 and C_2 are defined as the union of these discs. To compute the values for the union, the branch and bound method described in Section 8.7 was used. The number between brackets represents the number of relaxed problems that were solved during a run of the branch and bound procedure. A dash (-) represents infeasible cases, *i.e.* cases were X^1 is empty.

Radius	$\inf(X^1)$		$\sup(X^1)$	
a	Convex hull	Union	Convex hull	Union
0.001	0.246	- (13)	0.714	- (13)
0.01	0.230	0.447 (9)	0.729	0.729 (5)
0.05	0.159	0.219 (3)	0.782	0.779 (5)

oil is a common practice. Multiple reasons exist for this adulteration. In some cases, the adulteration leads to an oil blend with tractable properties (*e.g.* a good taste). However, mostly oils are adulterated for economic reasons. Indeed, when comparing the production costs of a pure high-quality vegetable oil with a blend of the high-quality (expensive) oil with a cheaper vegetable oil, one often finds a considerable difference in these costs. When vegetable oils are marketed, some level of adulteration θ can often be allowed[4]. However, due to the economic impact, these allowable levels are sometimes exceeded. To detect these fraudulent cases, a variety of methodologies and procedures has been developed throughout the past decades. These methods mainly differ in: the type of (chemical) information about the oil blends that is required, and the (statistical) methodology that is used to obtain an estimate of the level of adulteration.

The demonstration in this section fits within the general problem setting described above. More precisely, we will use our set-estimator to detect fraudulent levels of adulteration. We use a problem setting borrowed from [115] (notably, this problem setting is rather popular), where the authors use the fatty composition of vegetable oils to detect adulteration. In [115], the authors aim to detect adulteration of mustard seed oil with soybean oil. In their study, the authors collected (mainly from literature) more than 200 fatty acid profiles mostly of oils extracted from oil-producing *Brassica* species and soybean oil. Fatty acid profiles consist of the mass percentages of the following 13 fatty acids: C16:0, C16:1, C18:0, C18:1, C18:2, C18:3, C20:0, C20:1, C20:2, C22:0, C22:1, C24:0 and C24:1. In our experiments, we use these fatty acid profiles to represent the mixtures. To that end, all profiles of *Brassica* species and soybean oils (197 in total) were extracted. Moreover, following the original manuscript, the oils were classified in three groups: *Brassica* oils with a high erucic acid content (C22:1 content higher than 12 %, 138 observations),

[4] The allowed adulteration level is expressed as the relative amount of adulterating oil to the total amount of oil in a blend (in this work mass percentages).

Brassica oils with a low erucic acid content (37 observations) and soybean oils (22 observations).

9.3.2. Estimating the level of adulteration

The main goal consists of detecting, for a given oil blend, whether the adulteration level (the adulterating oil is soybean oil) is above a given threshold. Let the sources be *Brassica* oils with a high erucic acid content (source 1), *Brassica* oils with a low erucic acid content (source 2) and soybean oil (source 3). Moreover, let x_1, x_2 and x_3 be the proportional contributions of these sources to the oil blend with observed mixture vector $\mathbf{y}$. For a given threshold θ, this problem reduces to assessing whether $x_3 > \theta$. However, following the philosophy of our set estimator, such an assessment is often non-trivial (at least not without making additional assumptions). The representation of the (pure) oils (the sources) by means of their fatty acid profiles can be used to construct set-based descriptions of the three pure oils. Therefore, let C_i (with $i = 1, 2, 3$) be defined as a subset of $\mathbb{R}^{13}$ that contains the representation of all pure oils that belong to source i. Moreover, let $\mathbf{y}$ be the mixture vector of a given oil blend. For a fixed threshold θ, we know that

(i) If $\theta < \inf(X^3(\mathbf{y}, C_1, C_2, C_3))$ the adulteration level x_3 of $\mathbf{y}$ is above θ.

(ii) If $\theta > \sup(X^3(\mathbf{y}, C_1, C_2, C_3))$ the adulteration level x_3 of $\mathbf{y}$ is below θ.

(iii) If $\theta \in X^3(\mathbf{y}, C_1, C_2, C_3)$ it can not be concluded whether the adulteration level x_3 is above or below the threshold θ.

Naturally, the usefulness of this procedure strongly depends of the expressiveness of the sets C_1, C_2 and C_3. Indeed, when these sets contain little information that can be used to discriminate the components in the mixture, the interval $X^3(\mathbf{y}, C_1, C_2, C_3)$ can become too wide to be useful in practice.

Unfortunately, the sets C_1, C_2 and C_3 are unknown. Additionally, we are only given a (potentially noisy) observation $\tilde{\mathbf{y}}$ of the mixture vector $\mathbf{y}$. The data described before can be used to obtain estimates $\hat{C}_1$, $\hat{C}_2$ and $\hat{C}_3$ (for example by taking the convex hull). Moreover, the observed mixture vector $\tilde{\mathbf{y}}$ can be used as a substitute for $\mathbf{y}$. Subsequently, $X^3(\tilde{\mathbf{y}}, \hat{C}_1, \hat{C}_2, \hat{C}_3)$ can be used in the decision process above.

Hereafter, several experiments are described in which our set estimator is applied to obtain an interval estimate of the adulteration level of soybean oil in an oil blend. All experiments were performed on a standard desktop computer (Intel Dual Core with 4Gb RAM). Moreover, in the following experiments a single run of a solver never took more than 15 seconds to achieve convergence (except for the B&B procedure, which takes about 15 minutes to complete). It should be noted that all data were centered and scaled (to ensure a sample variance of one for each fatty acid).

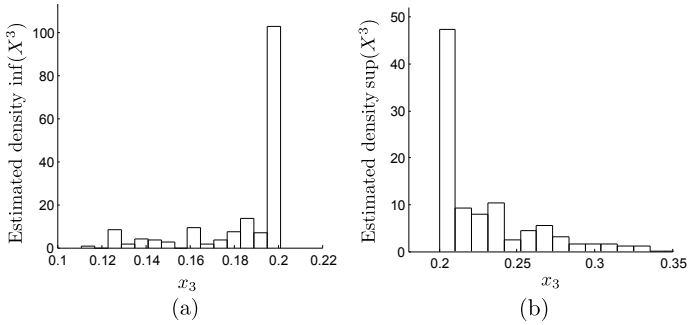


Figure 9.7: (a) Visualization of a sample (by means of a histogram) of $\inf(X^3)$ (sample size 200) for Experiment 1. (b) Visualization of a sample (by means of a histogram) of $\sup(X^3)$ (sample size 200) for Experiment 1.

Experiment 1. In a first experiment, the convex hulls of the fatty acid profiles (referred to as the observed prototype vectors) in the three datasets are used to estimate C_1, C_2 and C_3. To obtain a mixture vector $\mathbf{y}$, from each source, an observed prototype vector was randomly selected. Let $\mathcal{C}^1$, $\mathcal{C}^2$, $\mathcal{C}^3$ denote the random variables that represent the observed prototypes that were selected. Subsequently, the vector of mixing proportions is generated, by sampling from a uniform distribution over the simplex with the constraint that $x_3 = 0.2$. Let the random vector $\mathcal{X} = (\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3)$ represent these mixing proportions. $\mathbf{y}$ is now simulated using the LMM. Let $\mathcal{Y}$ be the random mixture vector:

$$\mathcal{Y} = \mathcal{X}_1 \mathcal{C}^1 + \mathcal{X}_2 \mathcal{C}^2 + \mathcal{X}_3 \mathcal{C}^3 .$$

The histograms in Figure 9.7 visualize a sample of the random variable $\inf(X^3(\mathcal{Y}, \hat{C}_1, \hat{C}_2, \hat{C}_3))$ and $\sup(X^3(\mathcal{Y}, \hat{C}_1, \hat{C}_2, \hat{C}_3))$ with a sample size of 200. The histograms in these figures can be interpreted as approximations of the probability density functions of the random variables they were sampled from.

Based on Figure 9.7 we argue that the set estimator is useful in cases that are representative for the experimental setup that was used. Indeed, when we choose $\theta = 0.25$ we can see from Figure 9.7(b) that the majority of the simulated blends (where the adulteration level was 0.2) will be considered non-fraudulent. In approximately 10% of the cases, the test will be inconclusive. Similar conclusions can be made when we choose $\theta = 0.15$. Here most (approximately 95%) of the simulated samples will be considered fraudulent.

Experiment 2. Unfortunately, the simulation study above is not representative for most real-life applications. There are at least two problems with the procedure above. Firstly, it is assumed that the observation of the mixture vector is free of noise. Secondly, the prototype vectors that are used to create a real-life mixture may not be included in the datasets. To investigate the influence of these shortcomings,

a second simulation experiment was conducted. The experimental setup of this experiment was identical to the one in Experiment 1, except for two modifications. Firstly, the noisy LMM was used instead of the original LMM. Consequently, we have that

$$\tilde{\mathcal{Y}} = \mathcal{X}_1 \mathcal{C}^1 + \mathcal{X}_2 \mathcal{C}^2 + \mathcal{X}_3 \mathcal{C}^3 + \mathcal{E}\,.$$

The random noise vector $\mathcal{E}$ was chosen to be multivariate normally distributed with a diagonal covariance matrix. Moreover, the variance of $\mathcal{E}_i$ was set to 5% of the variance of $\mathcal{Y}_i$. Secondly, the three observed prototype vectors that were used to construct $\tilde{\mathbf{y}}$ were not used in the estimation procedure of $\hat{C}_1$, $\hat{C}_2$ and $\hat{C}_3$.

In this experiment, for each of 200 repetitions, we obtained $X^3(\tilde{\mathbf{y}}, \hat{C}_1, \hat{C}_2, \hat{C}_3) = \emptyset$. Based on the discussion in Section 8.6 in the previous chapter, this should not come as a surprise. Indeed, the dimensionality q of the problem is 13. Comparing this dimensionality with the number of observed prototypes in class three (23 minus 1 observations), it can be expected that $\hat{C}_3$ (which is the convex hull of these observations) has a very low cardinality (when the cardinality is measured by means of the volume of $\hat{C}_3$). More strongly, it is not unlikely that $\hat{C}_3$ is (almost completely) part of a subspace of $\mathbb{R}^{13}$. As a result, it is unlikely that the extracted prototype vector will be an element of that convex hull. Consequently, we can expect that $X^3(\tilde{\mathbf{y}}, \hat{C}_1, \hat{C}_2, \hat{C}_3) = \emptyset$. Similar claims can be made regarding the influence of the error term.

Experiment 3. The 'problems' that were encountered in the previous experiment can be dealt with in two manners. Firstly, the sets $\hat{C}_1$, $\hat{C}_2$ and $\hat{C}_3$ can be extended using the ideas presented in the robust version of our set estimator. Secondly, following the discussion in Experiment 2, we could recognize that (based on the data that is available) the true dimensionality of the data is smaller than 13 and reduce the dimensionality. In this experiment, we take the former approach. In Experiment 4, the latter approach is considered.

In this experiment, we mimic the situation presented in Figure 9.4, with the exception that we use hyperspheres (with radius a) instead of ellipsoids. The radius a can be seen as parameter of our method. The choice of a is crucial to the usability of our set estimator. Indeed, when a is too large, the obtained intervals will be overly wide, whereas a value that is too small would lead to empty sets. Therefore, the following heuristic was used to estimate a. For each point in the dataset, the distance to its nearest neighbor was computed. From these distances, the maximum was selected. The parameter a was then set equal to this maximum ($a = 1.9$). The remaining settings of the experiment are identical to the settings in Experiment 2. It can easily be seen that this value for a will prevent X^3 from being empty. The histogram in Figure 9.8 visualizes a sample of the random variable $\sup(X^3(\mathcal{Y}, \hat{C}_1, \hat{C}_2, \hat{C}_3))$ with a sample size of 200. For each of the 200 runs, the infimum was zero. From Figure 9.8 it can be seen that the intervals that were obtained obtained are very wide. Unfortunately, the width of these intervals

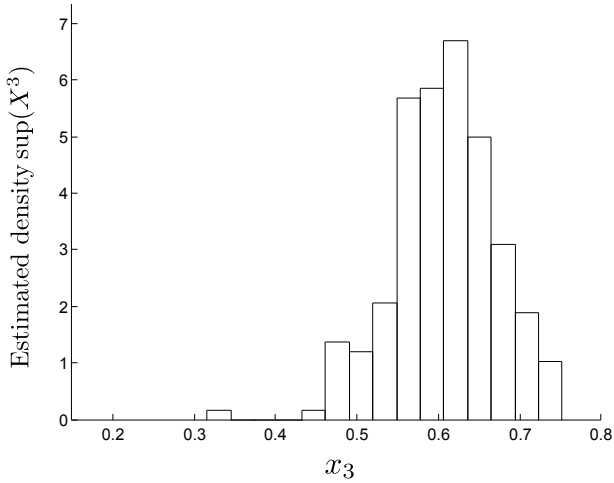


Figure 9.8: Visualization of a sample (by means of a histogram) of $\inf(X^3)$ (sample size 200) for Experiment 3, with radius $a = 1.9$.

strongly limits the possibility to detect fraudulent blends. Indeed, we can only state that most of the blends that were made have an adulteration level that is below 70%. As the true adulteration level was only 20%, this is a very weak result.

We argue that the results above cannot be attributed to the potential inadequacy of our method. On the contrary, these results illustrate that within this limited setting, we can only make very weak statements regarding the potential fraudulence of a given blend.

To reduce the width of the estimated intervals, we can reduce the size of the estimates $\hat{C}_1, \hat{C}_2, \hat{C}_3$. A simple way to achieve this consists of reducing the radius a of the hyperspheres. However, it should be noted that this reduction can lead to estimated intervals that potentially exclude the true adulteration level of 20%. We argue that the reduction of the radius implicitly adds assumptions to the estimation procedure. Naturally, when these assumptions do not hold, wrong conclusions can be drawn. Figure 9.9(a)–(b) shows histograms of the infima and the suprema of X^k for $a = 0.4$ (based on 200 runs). It should be noted that 11 of the runs lead to an empty interval, illustrating that the implied assumptions do not fully hold. Figure 9.9(a)–(b) (resp. 9.9(c)–(f)) shows histograms of the infima and the suprema of X^3 for $a = 0.4$ (resp. $a = 0.2$ and $a = 0.15$), based on 200 runs. From these figures, it can be seen that the estimated intervals narrow down. Unfortunately, the number of cases where $X^3 = \emptyset$ increases. For $a = 0.2$, 25 runs resulted in empty sets. For $a = 0.15$, this number increases to 54. Moreover, from Figure 9.9(e)–(f), it can be seen that for $a = 0.4, 0.2$ or 0.15 there are several cases where the true adulteration level of 20% is outside the estimated interval. The number of cases were $X^3 = \emptyset$ increases as the radius a decreases.

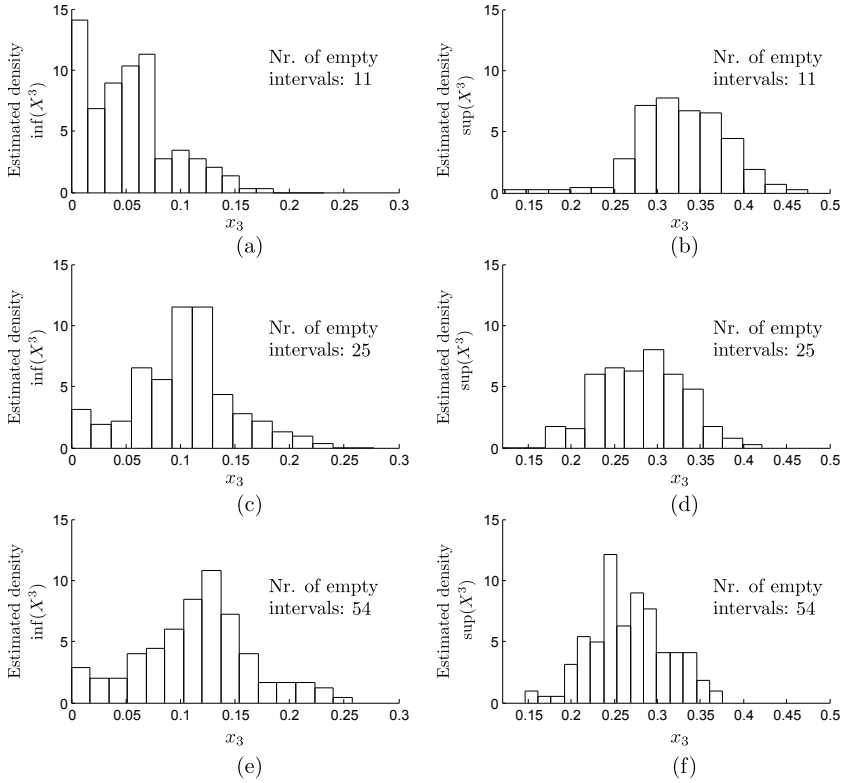


Figure 9.9: (a), (c), (e) Visualization of a sample (by means of a histogram) of $\inf(X^3)$ (sample size 200) for Experiment 3. (b), (d), (f) Visualization of a sample (by means of a histogram) of $\sup(X^3)$ (sample size 200) for Experiment 3. For (a) and (b) the radius $a = 0.4$. For (c) and (d) the radius is $a = 0.2$. For (e) and (f) the radius is $a = 0.15$.

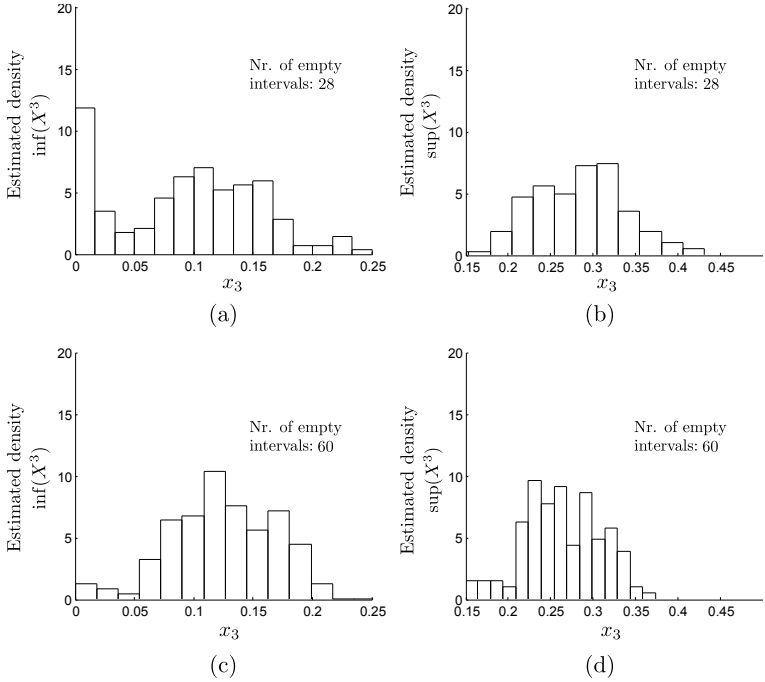


Figure 9.10: (a), (c), Visualization of a sample (by means of a histogram) of $\inf(X^3)$ (sample size 200) for Experiment 4. (b), (d), Visualization of a sample (by means of a histogram) of $\sup(X^3)$ (sample size 200) for Experiment 4. For (a) and (b) the number of retained principal components is $k = 6$. For (c) and (d) the number of retained principal components is $k = 8$.

Experiment 4. We now elaborate on the dimensionality reduction approach referred to in Experiment 3. Following the discussion in Section 8.6, we choose to reduce the dimensionality by only considering the first k principal component directions of the fatty acid data. Naturally, k is a parameter. Interestingly, the influence of k is similar to the influence of the radius a in Experiment 3. From several preliminary experiments[5], it was concluded that for $k < 6$, the intervals that are obtained are unusably wide. On the other hand, for $k > 9$, the number of 'mistakes', *i.e.* cases where $X^3 = \emptyset$ or $20\% \notin X^3$ is above 1/3. Figure 9.10(a)–(b) (resp. Figure 9.10(c)–(d)) shows histograms of the infima and the suprema of X^3 for $k = 6$ (resp. $k = 8$), based on 200 runs. It should be noted that, for $k = 6$ the number of runs where $X^3 = \emptyset$ was 28, whereas this number was 60 for $k = 8$.

Experiment 5. Following the discussion in Section 9.2.3 of this chapter, the intermediate blends that are implied in Experiments 3 and 4, may require a (unrealistically) large number of prototypes to be mixed. For example, when $k = 6$ in the previous experiment and using the geometric interpretation of the problem, the intermediate mixture may require up to 7 observed prototypes. To reduce

[5] The experimental setup was, except for the dimensionality reduction, identical to Experiment 2.

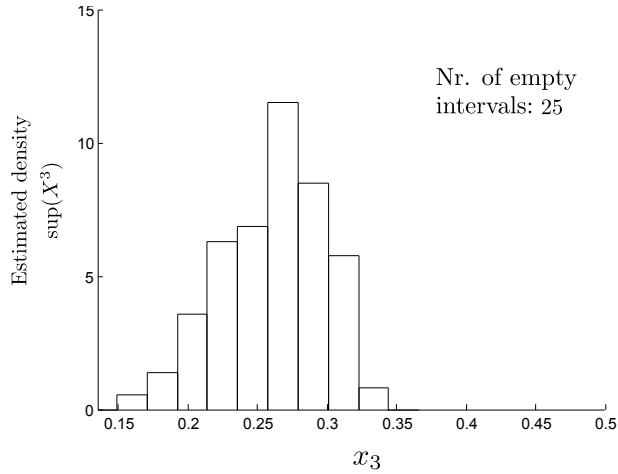


Figure 9.11: Visualization of a sample (by means of a histogram) of $\sup(X^3)$ (sample size 200) for Experiment 5.

this number, the sparse estimator is used in this experiment. The experimental setting is identical to the one used in Experiment 3 (except for the use of the sparse estimator). We choose $a = 0.3$. This value is slightly higher than the optimal one found in Experiment 3. However, due to the additional constraints implied by the sparse estimator (resulting from taking the union in stead of the convex hull), this relaxed form was preferred to reduce the number of cases where $X^3 = \emptyset$.

Figure 9.11 shows a histogram of $\sup(X^3)$, obtained using the sparse estimation procedure. It can be seen, as compared to the previous experiments, that the suprema that are obtained here are slightly lower. Nevertheless, only few cases occurred where the true adulteration rate of 20 % was lower than the estimated supremum. Therefore, we can conclude that the sparse estimator is an interesting estimator in this setting. Unfortunately, the branch and bound routine that is needed to compute this interval has a computational demand that is considerably higher than that of the other approaches. Solving the optimization problems in Experiments 1–4 takes only a few seconds on a desktop computer, whereas the branch and bound procedure takes approximately 10 minutes to complete.

9.4. Conclusions and discussion

In this chapter, the use of set-based unmixing procedures was illustrated by means of an experiment on artificial data as well as a real-life case study (Objective III.5). From the experiments on artificial data, it is clear that the interval estimator complements the probability distribution estimator and the point estimator. Moreover, it was illustrated that the robust and sparse variants of the interval

estimator clearly lead to different estimated intervals. As expected, in medium-to-high-dimensional settings (such as the case study) the robust and sparse variants are strongly influenced by the radii (the parameter a) that is used.

From the case study, it can be concluded that (at least in this case), the interval estimates that are obtained can be useful in practice to detect fraudulent levels of adulteration in some cases. Nevertheless, from the width of the histograms in Figures 9.9–9.11 it can be concluded that, when the true adulteration level only mildly exceeds the allowed adulteration level, the approach will often lead to a decision which is inconclusive. Depending on the application, this can be considered as problematic in some cases. However, this potential shortcoming cannot fully be attributed to the estimation procedure. On the contrary, we argue that the width of the intervals is the result of a lack of information in the data. As our interval estimator only makes very few assumptions, it heavily depends on the information that is present in the data. On the other hand, probability density estimators (and the resulting intervals) more heavily depend on assumptions. Naturally, these assumptions lead to more narrow intervals. However, as argued in Chapter 7, the assumptions that are typically made by probability distribution estimators are hard to verify. Therefore, the resulting intervals may be highly inaccurate in some cases. Moreover, the interpretation that can be given to these intervals often differs from the interpretation that is needed for the application at hand.

It seems that the requirement of sparsity in the solution can lead to more informative (more narrow) intervals. Interestingly, this can be related to the previous paragraph as requiring sparsity can be seen as an implicit way of making assumptions. Indeed, a sparse estimate implies the assumption that the mixture was created by blending a small number of prototypes. When those assumptions are violated, the resulting estimated interval may not contain the 'true' mixing proportion. Nevertheless, the assumption that is made here has a clear interpretation. Consequently, the intervals that are obtained here allow a clear interpretation as well. In Experiment 5, the mixtures were simulated using prototype vectors that were excluded from the data that was used to estimate the sets C_1, C_2 and C_3. This means that the assumption of sparsity (*i.e.* mixtures can be made using only a limited number of prototypes) is potentially violated. Nevertheless, the resulting intervals seem to be informative. This illustrates that the requirement of sparseness seems to be robust towards slight violations of the assumptions.

PART IV

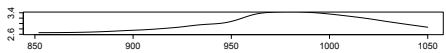
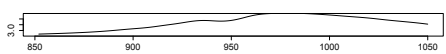
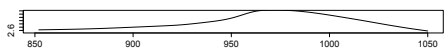
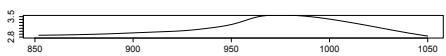
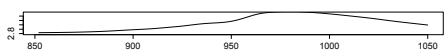
LEARNING TO PREDICT COMPOSITIONS

10 Learning to predict compositions: a machine learning problem

10.1. Introduction

As a starting point of this section, we return to the (introductory) problem setting of Chapter 2, involving the prediction of the relative amount of water, fat and oil in a minced meat sample based on the Near Infra Red (NIR) spectrum of that sample (see Table 10.1). This problem can be classified as a prediction problem. More precisely, we are interested in constructing a function that maps a NIR spectrum to a 3-part composition. Stated differently, the meat sample can be represented in two manners. Firstly, it can be represented by the proportional contributions of a set of components. Secondly, it can be represented by means of its NIR spectrum. The prediction problem can then be seen as the problem of finding a link between the two representations.

Table 10.1: Analysis results of five minced meat samples. The first column shows the sample number (conform the original dataset). The second (resp. third and fourth) column shows the relative amount (mass percentages) of water (resp. fat and protein) in the sample. The masses are expressed relative to the total amount of water, fat and protein in the sample. The fifth column shows a graphical display of the NIR-spectra of these samples (wavelength (nm) versus $\log(1/R)$). This dataset is a subset of a publicly available dataset known as the *tecator* dataset [2]. The complete dataset contains 240 instances.

nr	water	fat	protein	NIR spectrum
1	0.607	0.226	0.167	
2	0.462	0.403	0.135	
3	0.711	0.084	0.205	
4	0.732	0.059	0.209	
5	0.587	0.257	0.156	

Probably, if it even exists, it would be extremely hard to find a perfect link between the NIR spectrum of a sample and its composition. This means that it is unlikely that a function can be found that is capable of predicting the composition of any given sample using only its NIR spectrum without making mistakes. Mostly, the

predicted composition of a sample will not be identical to the true composition. As a result, we will need to tolerate some prediction errors. Naturally, functions that lead to small prediction errors should be preferred over functions that lead to large errors.

Within the scope of this dissertation, the (rather trivial) observations above allow the prediction problem to be recast as an optimization problem. Indeed, from a set of functions that map a NIR spectrum to a 3-part composition (the search space), we want to select the function that minimizes the prediction error (the objective function). Naturally, to transform this loosely described problem into a formal mathematical optimization problem, several choices need to be made regarding the set of functions that is considered (*e.g.* affine functions versus non-linear functions) and the error measure that is used.

The problem description given above perfectly fits within the field of *machine learning*. Consequently, in Part IV of this dissertation, we describe how several advances in the field of machine learning can be used to predict compositions. The main focus is twofold. We focus on translating the prediction problem into a formal optimization problem as well as on solving the resulting optimization problem in a variety of problem settings. The remainder of this chapter is organized as follows:

- In Section 10.2, we give a brief introduction to statistical learning theory (the experienced reader may safely skip this section).
- In Section 10.3, several strategies are presented that can be used to construct predictive models with compositional outputs.
- In Section 10.4, we experiment with the strategies that were presented in Section 10.3.
- In Section 10.5, conclusions are presented and discussed.

10.2. A brief introduction to statistical learning theory

This introductory section groups together several well-known results that can be found in textbooks[1] such as *Statistical Learning Theory* [116], *Pattern Recognition and Machine Learning* [55], *Learning with Kernels* [117] or *Modern Multivariate Statistical Techniques* [118]. It should be noted that this introduction is far from complete. We mainly focus on the principles that will be used later on in this dissertation. This introductory section is mainly intended for the reader who is

[1] The textbooks referred to hereafter are well known textbooks in the field of machine learning. Much of the insights that I (the author of this dissertation) have acquired throughout the years preceding the submission of this dissertation stem from reading those textbooks.

unfamiliar with statistical learning theory, regularization and hypothesis spaces. The experienced reader can safely skip this section.

10.2.1. The basis of statistical learning theory

Statistical learning theory[2] provides a theoretical basis for (the automation of) inductive inference processes. An inductive inference process mainly consists of three phases: (1) the observation of a phenomenon; (2) the construction of a model for that phenomenon; and (3) the use of that model to make predictions. Naturally, this process is described rather loosely. Statistical learning theory formalizes this process. As we have seen several times throughout this dissertation, a formalization often entails several choices, simplifications and assumptions. We elaborate on these issues hereafter.

As we will only be dealing with prediction problems, we limit this discussion to that type of problems. In the first phase, the observation of a phenomenon gives rise to a (multi)set of input-output pairs. In the most general case, the inputs are elements of some input set X. We do not make any assumptions regarding the structure[3] of X. In a similar way, the outputs are elements of a set Y. As for X, no additional assumptions are required on Y in the most general case. However, we limit this discussion to situations where Y is a subset of $\mathbb{R}^q$. We generally denote an input-output pair as $(\mathbf{x}, \mathbf{y}) \in X \times Y$. Statistical learning theory assumes that there exists a probabilistic model that is capable of completely describing the observed phenomenon. This means that input-output pairs can be modeled as observations of some random vector $(\mathcal{X}, \mathcal{Y})$ whose sample space is $X \times Y$. Therefore, the link between several input-output pairs is that they are independent observations of the same random vector (we say that these observations are i.i.d. or independently and identically distributed). Notably, the probabilistic model implies that, given an input $\mathbf{x}$, there is not a single corresponding output, but a multitude of outputs with conditional probability distribution function $\rho_{\mathcal{Y}|\mathcal{X}}(\cdot \mid \mathbf{x})$.

We now proceed to the modeling step of the inductive process. Note that a random vector $(\mathcal{X}, \mathcal{Y})$ is completely described by its probability density function $\rho_{\mathcal{X},\mathcal{Y}}$. Therefore, in the modeling phase, it can be attempted to induce the unknown probability distribution $\rho_{\mathcal{X},\mathcal{Y}}$ using the observations that were made in the first phase. However, for prediction problems, the approximation of $\rho_{\mathcal{X},\mathcal{Y}}$ is not the ultimate goal. Instead, we are interested in finding a function $\mathbf{f} : X \to Y$ that can be used to predict the output that corresponds to an observed input. Therefore, the result of the modeling phase (phase 2) is such a function. As argued before, the underlying probabilistic model implies that a single input $\mathbf{x}$ often can be linked

[2] The starting point of this section is inspired on [119]. A more complete discussion of some of the principles described here can be found therein.

[3] In traditional examples, X is often a subset of $\mathbb{R}^p$. However, X can as well be a set of graphs, pictures, ...

with a multitude of outputs. As we want to learn a deterministic function $\mathbf{f}$, we will need to tolerate some errors. More precisely, given an observation $(\mathbf{x}, \mathbf{y})$, it is likely that $\mathbf{f}(\mathbf{x}) \neq \mathbf{y}$. Naturally, it is desirable that the dissimilarity between $\mathbf{f}(\mathbf{x})$ and $\mathbf{y}$ is as small as possible. Therefore, let $\ell : Y \times Y \to \mathbb{R}^+$ be a function that measures the dissimilarity between two elements of Y such that $\ell(\mathbf{y}, \mathbf{y}) = 0$ for all $\mathbf{y} \in Y$. We typically call this function a *loss function.* Given an input vector $\mathbf{x}$, the loss function ℓ can be used to define the *risk* associated with a prediction $\hat{\mathbf{y}}$:

$$\int_Y \ell(\hat{\mathbf{y}}, \mathbf{y})\, \rho_{\mathcal{Y}|\mathcal{X}}(\mathbf{y} \mid \mathbf{x})\, \mathrm{d}\mathbf{y}\,. \tag{10.1}$$

In words, this formula simply measures the expected value of the loss function when $\hat{\mathbf{y}}$ is used to predict the output of an observed object with input vector $\mathbf{x}$. Subsequently, writing the formula above as a function of $\mathbf{y}$, the optimal prediction $\mathbf{y}^*$ can be defined as

$$\mathbf{y}^* = \arg\min_{\hat{\mathbf{y}}} \int_Y \ell(\hat{\mathbf{y}}, \mathbf{y})\, \rho_{\mathcal{Y}|\mathcal{X}}(\mathbf{y} \mid \mathbf{x})\, \mathrm{d}\mathbf{y}\,,$$

where we assume that the minimizer is unique. Similarly, the 'optimal' prediction function $f^* : X \to Y$ is defined as:

$$\mathbf{f}^*(\mathbf{x}) = \arg\min_{\hat{\mathbf{y}}} \int_Y \ell(\hat{\mathbf{y}}, \mathbf{y})\, \rho_{\mathcal{Y}|\mathcal{X}}(\mathbf{y} \mid \mathbf{x})\, \mathrm{d}\mathbf{y}\,. \tag{10.2}$$

Interestingly, Eq. (10.2) completely defines the optimal prediction function $\mathbf{f}^*$. However, its computation requires the conditional distribution of $\mathcal{Y}$ given $\mathcal{X} = \mathbf{x}$ to be known for every value of $\mathbf{x}$. Unfortunately, in most cases, this distribution is unknown. Instead, we are only given a finite sample of the random vector $(\mathcal{X}, \mathcal{Y})$. We generally denote that sample (with sample size n) as $T = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$. The (multi)set T is often called the training (data)set, as it can be used to build or train a predictive model. Two options exist to obtain an approximation of $\mathbf{f}^*$ using dataset T. We elaborate on these options hereafter.

Firstly, T can be used to estimate the conditional distribution of $\mathcal{Y}$ given $\mathcal{X} = \mathbf{x}$. Subsequently, this estimate can be used to replace the conditional distribution in Eq. (10.2). This approximation problem is generally known as a density estimation problem. For example, a histogram of the observations could be used here to estimate $\rho_{\mathcal{Y}|\mathcal{X}}$ (in some cases).

As a second option, a set of functions H (generally called the hypothesis space) is defined. For example, H can be the set of affine functions with domain X and co-domain Y. The risk of a function $\mathbf{f} \in H$ is defined as:

$$r(\mathbf{f}) = \int_{X \times Y} \ell(\mathbf{f}(\mathbf{x}), \mathbf{y})\, \rho_{\mathcal{X},\mathcal{Y}}(\mathbf{x}, \mathbf{y})\, \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{y}\,, \tag{10.3}$$

Now, let $\tilde{f} \in H$ be the function that minimizes this risk, *i.e.*

$$\tilde{\mathbf{f}}(\mathbf{x}) = \arg\min_{\mathbf{f}\in H} \int_{X\times Y} \ell(\mathbf{f}(\mathbf{x}), \mathbf{y})\, \rho_{\mathcal{X},\mathcal{Y}}(\mathbf{x}, \mathbf{y})\, \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{y} \tag{10.4}$$

$$= \arg\min_{\mathbf{f}\in H} \int_{X\times Y} \ell(\mathbf{f}(\mathbf{x}), \mathbf{y})\, \rho_{\mathcal{Y}|\mathcal{X}}(\mathbf{y} \mid \mathbf{x})\, \rho_{\mathcal{X}}(\mathbf{x})\, \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{y} \tag{10.5}$$

Here, Eq. (10.5) shows the link between Eqs. (10.4) and (10.2). Eqs. (10.3) and (10.4) still require the joint distribution of $(\mathcal{X}, \mathcal{Y})$ to be known. However, the set T can be used to estimate the risk of a function $\mathbf{f} \in H$ (given in Eq. (10.3)) as follows:

$$r_{\mathrm{e}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\mathbf{x}_i), \mathbf{y}_i)\,. \tag{10.6}$$

This approximation is called the *empirical risk*. The minimizer $\hat{\mathbf{f}} \in H$ of the empirical risk can be used to obtain a function that can be used to make predictions:

$$\hat{\mathbf{f}}(\mathbf{x}) = \arg\min_{\mathbf{f}\in H} \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{f}(\mathbf{x}_i), \mathbf{y}_i)\,.$$

Hypothesis spaces and regularization

In principle, the risk function r scores the functions in H. Subsequently, the element of H that has minimal risk can be selected as the optimal prediction function. However, as the risk function requires knowledge of the joint distribution, it cannot be used in practice. Instead, the empirical risk is used. Therefore, it is hoped that the element of H that minimizes the empirical risk r_{e} also (approximately) minimizes the risk r, *i.e.* $\tilde{\mathbf{f}}$ is similar to $\hat{\mathbf{f}}$. Unfortunately, when the size of H is large, it is likely that H will contain functions $\mathbf{f}$ for which $r_{\mathrm{e}}(\mathbf{f}) < r_{\mathrm{e}}(\tilde{\mathbf{f}})$. In those cases, the minimizer of r_{e} may not be optimal. On the other hand, when H is too small, it is likely that $\tilde{\mathbf{f}}$ and $\mathbf{f}^*$ are very dissimilar.

As a generic approach, we could initially choose H to be a rather large hypothesis space and, subsequently from this space exclude all 'unlikely' functions (*i.e.* functions for which it is believed that they are bad candidates for building a predictive model). Theoretical as well as applied work has demonstrated that the elimination of extremely irregular candidates from H is often beneficial. Therefore, given a function $c : H \to \mathbb{R}^+$ that measures the irregularity of the elements of H, and a scalar $\lambda > 0$ we can use $\{\mathbf{f} \in H \mid c(\mathbf{f}) \leq \lambda\}$ as a 'restricted' hypothesis space. This leads to the following constrained estimator:

$$\arg \min_{\mathbf{f}\in\{\mathbf{g}\in H \mid c(\mathbf{g})\leq\lambda\}} \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{f}(\mathbf{x}_i), \mathbf{y}_i)\,. \tag{10.7}$$

In the formulation above, highly irregular functions were excluded from the hypothesis space. Alternatively, the selection of highly irregular (or complex) functions can be avoided by using a so-called regularized empirical risk:

$$r_{\mathrm{e}_\lambda}(\mathbf{f}) = \frac{1}{n}\sum_{i=1}^{n}(\ell(f(\mathbf{x}_i), \mathbf{y}_i)) + \lambda\, c(f)\,. \tag{10.8}$$

We now define

$$\hat{\mathbf{f}}_\lambda = \arg\min_{\mathbf{f}\in H} r_{\mathrm{e}_\lambda}(\mathbf{f})\,.$$

Interestingly, there exist a strong relationship between the minimizer in Eq. (10.7) and $\hat{\mathbf{f}}_\lambda$.

It is generally known that when a regularized empirical risk function is used to learn a function that can be used for prediction, there exists a trade-off between the fit of the function to the data (expressed by the empirical loss) and the complexity or regularity of the function (expressed by the magnitude of the norm). Therefore, the term $\lambda\, c(\mathbf{f})$ of the regularized empirical risk is often called the regularization term. λ is called the regularization parameter. In summary, the regularized empirical risk function will favor functions that fit reasonably well to the data and have a low complexity.

Remark. In this section, it was argued that the goal of a predictive modeling problem exists in finding a function f that minimizes the risk $r(\mathbf{f})$. Naturally, the loss function that is choosen to define r can have a strong impact on the result. As the computation of $r(\mathbf{f})$ is infeasible in practice, it was argued that the (regularized) empirical risk could be used as a criterion to select a function. The discussion above may suggest that the loss function that is used in the definition of the risk should be identical to the one that is used to define the (regularized) empirical risk. However, this is not necessarily the case. There exist plenty examples in which a better predictive model can be obtained by using a loss function for the empirical risk that differs from the loss function used to define the risk.

10.2.2. Statistical learning as an optimization problem

Interestingly, the statistical learning framework presented above can be seen as a (rigorous) optimization problem. Indeed, predictive modeling problems often start with a loosely described problem setting such as "*find a function that optimally predicts the output variable for new observations*". To formalize this problem, statistical learning theory assumes that there exists a probabilistic model that fully describes the observed phenomenon. To arrive at a formal optimization problem, several other choices need to be made:

(i) A loss function $\ell : X \times X \to \mathbb{R}^+$ has to be defined.

(ii) The hypothesis space H needs to be defined.

(iii) A regularity criterion $c : H \to \mathbb{R}$ has to be defined.

(iv) A value for the regularization parameter λ has to be selected.

Given these ingredients, the regularized empirical risk function can be used to translate the learning problem into a formal mathematical optimization problem:

$$\underset{f \in H}{\text{minimize}} \quad \sum_{i=1}^{n} \ell(f(\mathbf{x}_i), \mathbf{y}_i) + \lambda\, c(f)\,.$$

The procedures that can be used to solve the optimization problem above strongly depend on the choice of the hypothesis space as well as on the loss function used. For example, let $X = \mathbb{R}^p$ and $Y = \mathbb{R}$. In that case the learning problem is a traditional regression problem. Consequently, we can for instance choose the squared loss function: $\ell(y, \hat{y}) = (y - \hat{y})^2$. Moreover, when we choose H as the space of all affine functions that map X to Y, each $f \in H$ can be identified with a vector $(\mathbf{a}^\top, b) \in \mathbb{R}^{p+1}$ such that $f(\mathbf{x}) = \mathbf{a}^\top \mathbf{x} + b$ for all $\mathbf{x} \in X$. Here, the irregularity of a function $f(\mathbf{x}) = \mathbf{a}^\top \mathbf{x} + b$ can be measured by the squared Euclidean norm (also called the L_2-norm) of $\mathbf{a}$, *i.e.* $c(\mathbf{f}) = \|\mathbf{a}\|_2^2$. When combining all these ingredients, the following optimization problem is obtained:

$$\mathcal{M}_1 : \underset{(\mathbf{a},b) \in \mathbb{R}^p \times \mathbb{R}}{\text{minimize}} \quad \sum_{i=1}^{n} (\mathbf{a}^\top \mathbf{x}_i + b - y_i)^2 + \lambda\, \|\mathbf{a}\|_2^2\,.$$

The strategy described above is known as ridge regression [120] and is one of the most popular regression methods used in practice.

Naturally, there exist other loss function that can be used instead of the squared loss. A popular alternative is the ϵ-insensitive loss function: $l(y, \hat{y}) = \max(0, |y - \hat{y}| - \epsilon)$. The resulting strategy is known as support vector regression [116]:

$$\mathcal{M}_2 : \underset{(\mathbf{a},b) \in \mathbb{R}^p \times \mathbb{R}}{\text{minimize}} \quad \sum_{i=1}^{n} \max\left(0, \left|\mathbf{a}^\top \mathbf{x}_i + b - y_i\right| - \epsilon\right) + \lambda\, \|\mathbf{a}\|_2^2\,.$$

Alternatively, we could also use the L_1-norm of the parameter vector $\mathbf{a}$ as a complexity measure, *i.e.* $c(f) = \|\mathbf{a}\|_1$ to obtain the following optimization problem:

$$\mathcal{M}_3 : \underset{(\mathbf{a},b) \in \mathbb{R}^p \times \mathbb{R}}{\text{minimize}} \quad \sum_{i=1}^{n} (\mathbf{a}^\top \mathbf{x}_i + b - y_i)^2 + \lambda\, \|\mathbf{a}\|_1\,.$$

This strategy is generally known as lasso regression [121].

From an optimization point of view, $\mathcal{M}_1$ can be solved as an unconstrained smooth and convex optimization problem. Alternatively, there exists a closed form solution of $\mathcal{M}_1$, however, this method requires the inversion of a $p \times p$ matrix, which can be computationally demanding. $\mathcal{M}_2$ can be transformed into an equivalent linearly constrained convex quadratic optimization problem [117] that can be solved using standard optimization software. Lastly, $\mathcal{M}_3$ can be transformed into an equivalent second order cone program [10].

10.2.3. Concluding remarks

In this section, the general framework of statistical learning theory was briefly described. In part IV of this dissertation, we will often be relying on the theory that was reviewed in this introductory section.

10.3. Predictive modeling of compositional data

Surprisingly, there exists only a very limited literature on the predictive modeling of compositional data. Most related work that deals with the modeling of compositional data focuses on statistical modeling [1, 5]. In those studies, modeling is mainly used in the scope of statistical hypothesis testing. As a result, issues involving regularization and high-dimensional data have (as far as we know) not been dealt with. Therefore, in the following sections, we present several approaches that can be adopted.

10.3.1. Approaches to predictive modeling of compositional data

As a starting point of this section, recall the problem setting presented in Section 10.1, where a predictive model needs to be constructed that can be used to predict the composition of a meat sample from its NIR spectrum. Clearly, the output space is $Y = \mathbb{S}^3$. On the other hand, the input space X is the space of all NIR spectra. For simplicity, we assume that $X = \mathbb{R}^p$. As a result, the learning phase should result in a function $\mathbf{f} : \mathbb{R}^p \to \mathbb{S}^q$ (with $q = 3$). Therefore, the hypothesis space H can be defined as the set of functions $\mathbf{f} : \mathbb{R}^p \to \mathbb{S}^3$ that can be written in the following form:

$$
\begin{aligned}
f_i(\mathbf{x}) &= \frac{\exp(\mathbf{a}_i^\top \mathbf{x} + b_i)}{1 + \sum_{i=1}^{q-1} \exp(\mathbf{a}_i^\top \mathbf{x} + b_i)}\,, \qquad \text{for } i = 1, \ldots, q-1\,, \\
f_q(\mathbf{x}) &= \frac{1}{1 + \sum_{i=1}^{q-1} \exp(\mathbf{a}_i^\top \mathbf{x} + b_i)}\,,
\end{aligned}
\tag{10.9}
$$

where $\mathbf{a}_i \in \mathbb{R}^p$ and $b_i \in \mathbb{R}$. Moreover, this formulation allows for the following complexity criterion $c(\mathbf{f}) = \sum_{j=1}^{q-1} \|\mathbf{a}_j\|_2^2$.

Lastly, a suitable loss function needs to be choosen. As the output is a vector of real values, the component-wise squared loss function could be selected, *i.e.*

$$\ell_{\mathrm{s}}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{j=1}^{q} (y_j - \hat{y}_j)^2 = \|\mathbf{y} - \hat{\mathbf{y}}\|_2^2 \,.$$

This leads to the following optimization problem:

$$\underset{(\mathbf{a}_j, b_j)_{j=1}^{q-1} \in \mathbb{R}^{(p+1)\times(q-1)}}{\text{minimize}} \quad \sum_{i=1}^{n} \|\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i\|_2^2 + \lambda \sum_{j=1}^{q-1} \|\mathbf{a}_j\|_2^2 \,. \tag{10.10}$$

Unfortunately, this problem is highly non-convex. Therefore, we can only attempt to solve it locally. Additionally, the exponential terms can result in numerical instability. As an alternative, we could attempt to use (a variant of) the multinomial deviance [122] as a loss function, *i.e.*

$$\ell_{\mathrm{d}}(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{j=1}^{q} y_j \log(\hat{y}_j) \,.$$

The use of this loss function leads to the following optimization problem:

$$\underset{(\mathbf{a}_j, b_j)_{j=1}^{q-1} \in \mathbb{R}^{(p+1)\times(q-1)}}{\text{minimize}} \quad \sum_{i=1}^{n} \sum_{j=1}^{q} y_{i,j} \log(f_j(\mathbf{x}_i)) + \lambda \sum_{j=1}^{q-1} \|\mathbf{a}_j\|_2^2 \,. \tag{10.11}$$

It can be shown that the optimization problem above is convex (see for instance [10]).

With respect to the discussion in the introductory chapter on compositional data analysis, it may be argued that a loss function that is used to measure the dissimilarity between two compositions should respect the main principles of compositional data analysis. Therefore, we could use the squared Aitchison distance as a loss function, *i.e.*

$$\ell_{\mathrm{a}}(\mathbf{y}, \hat{\mathbf{y}}) = d_{\mathrm{a}}(\mathbf{y}, \hat{\mathbf{y}})^2 \,.$$

This loss function can be used to construct another optimization problem. However, recall from Chapter 2 that $d_{\mathrm{a}}(\mathbf{y}, \hat{\mathbf{y}})^2 = \|\mathfrak{i}_E(\mathbf{y}) - \mathfrak{i}_E(\hat{\mathbf{y}})\|_2^2$, where $\mathfrak{i}_E$ represents the isometric log-ratio transform with orthonormal basis E. Therefore, we can use the (more familiar) squared loss in the transformed space instead of the Aitchison distance. Moreover, as the sample space of $\mathfrak{i}_E(\mathbf{y})$ is $\mathbb{R}^{q-1}$ (as opposed to $\mathbb{S}^q$), the hypothesis space H can be the space of affine vector functions:

$$H = \{f : \mathbb{R}^p \to \mathbb{R}^{q-1} \mid (\exists(\mathbf{A}, \mathbf{b}) \in \mathbb{R}^{(q-1)\times p} \times \mathbb{R}^{q-1})(\forall \mathbf{x} \in \mathbb{R}^{q-1})(\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b})\}\,.$$

Combining these ingredients, we obtain a multivariate regularized least squares problem (also called a multiple output least squares problem) [118]:

$$\underset{(\mathbf{A},\mathbf{b})\in\mathbb{R}^{(q-1)\times p}\times\mathbb{R}^{q-1}}{\text{minimize}} \quad \sum_{i=1}^{n} \|\mathbf{A}\mathbf{x}_i + \mathbf{b} - \mathbf{i}_E(\mathbf{y}_i)\|_2^2 + \lambda \sum_{j=1}^{q-1} \|\mathbf{A}_{j,\cdot}\|_2^2\,. \tag{10.12}$$

Let $(\hat{\mathbf{A}}, \hat{\mathbf{b}})$ be the optimal point of this optimization problem. Naturally, the function $\hat{\mathbf{f}}(\mathbf{x}) = \hat{\mathbf{A}}\mathbf{x} + \hat{\mathbf{b}}$ can not be used directly to predict the composition of new mixtures. Instead, we need to use the inverse log-ratio transform: $\mathbf{i}_E^{-1}(\hat{\mathbf{f}}(\mathbf{x}))$.

10.3.2. Selecting a loss function

The discussion in the previous sections may raise the question on how to select a particular hypothesis space, loss function and complexity term. Unfortunately, there is no direct answer to this question. A lot depends on the application to which the methodology is applied. With respect to the choice of a particular loss function, it is advisable to start by choosing an application-oriented loss function to define the risk r. For example, if an application requires a focus on the relative information contained in the compositional vector, then it may be worth considering the Aitchison distance as a loss function. On the other hand, if the focus is on the absolute, rather than the relative information (even though this may be questionable), the squared loss could be used.

Secondly, when selecting a loss function that will be used to construct the (regularized empirical risk) objective of an optimization problem, different criteria need to be considered (see also Remark 1). In particular, properties of the data such as the size of the dataset, the (assumed) presence of outliers, the complexity of the resulting optimization problem, and the loss function used to define r can be taken into account.

To gain insight into some of the differences between the three loss functions proposed earlier, we will now consider the case where $q = 2$. Figure 10.1 shows the loss functions for the isometric tranformation (ℓ_{a}), deviance (ℓ_{d}) and least-squares (ℓ_{s}) loss functions. This figure illustrates the clear difference between the three loss functions. In contrast to ℓ_{s}, both ℓ_{a} and ℓ_{d} are unbounded loss functions that become strongly asymmetric as the values of the labels tend to 0 or 1. Moreover, ℓ_{a} becomes very steep at the boundaries of $[0, 1]$. This property makes the learning

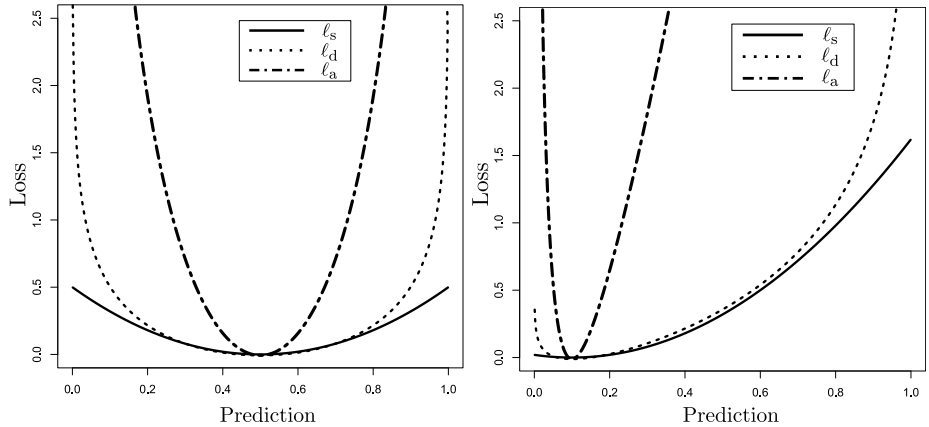


Figure 10.1: A visualization of the loss functions $\ell(\mathbf{y}, .)$: the isometric transformation with least squaes (ℓ_a), deviance (ℓ_d) and least-squares (ℓ_s) loss functions. The left and right figure give the loss when the true label vectors are $\mathbf{y} = (0.5, 0.5)$ and $\mathbf{y} = (0.1, 0.9)$, respectively (note that $\ell_d(\mathbf{y}, \mathbf{y}) \neq 0$, therefore the panels show $\ell_d(\mathbf{y}, .) - \ell_d(\mathbf{y}, (0.5, 0.5))$, resp. $\ell_d(\mathbf{y}, .) - \ell_d(\mathbf{y}, (0.1, 0.9))$).

procedure that uses ℓ_a sensitive to outliers or noisy datapoints located at the endpoints of the simplex. Consequently, we could argue that ℓ_d and ℓ_s are more robust loss functions, which could result in more stable models.

10.3.3. Selecting the hypothesis space

In the examples presented earlier in this chapter, we mainly used a hypothesis space of affine (vector) functions (for example in $\mathcal{M}_1$) or transformations of affine functions (for instance Eq. (10.9)). Naturally, several alternatives exist here. For example, as an extension of a hypothesis space of affine functions, we could select a space of all mth order polynomials or a spline basis. Alternatively, the hypothesis space could consist of all possible regression trees [123, 124]. Prior knowledge could be used to make such a selection. We elaborate on the selection of the hypothesis space in the following chapters.

10.3.4. Selecting a complexity criterion

In Section 10.2.2, a complexity criterion was introduced as a measure of the irregularity of a function. However, in a more general form, it can be seen as a manner to express a prior preference towards some functions in the hypothesis space. Therefore, the learning process is biased towards functions that are preferred by the complexity criterion. For example, the L_2-norm that is used in $\mathcal{M}_1$ and $\mathcal{M}_2$ will result in a function that is biased towards $\mathbf{0}_p$. As a result, this complexity criterion

introduces some prior belief that the distance between the optimal parameter vector and $\mathbf{0}_p$ is not too large. Naturally, if this prior belief is correct, this approach will improve the performance.

This reasoning suggests that different types of regularization terms can be used to include different types of of prior belief or prior knowledge. This interpretation of the regularization is very popular in a Bayesian approach to predictive modeling [55]. In the following chapter, different regularization strategies will be applied to the predictive modeling of compositional data.

10.4. An experimental study of loss functions for compositional data

In Subsections 10.3.1 and 10.3.2, we made several claims about three loss functions (ℓ_{a}, ℓ_{s} and ℓ_{d}) that can be used for learning predictive models for compositional data. In this section, we experiment with the *tecator* dataset (presented in Section 10.1) to investigate these claims. More precisely, in this section, predictive models are learned using these loss functions. The estimation procedures are described hereafter.

- Minimizing ℓ_{s}: the regularized empirical risk function described in Eq. (10.10) is minimized. However, instead of using models that are inherently linear functions of the inputs, the hypothesis space is extended by allowing any third order polynomial of the inputs. The complexity parameter λ is tuned using a tuning set (which is a subset of the data used to train the model).
- Minimizing ℓ_{d}: the regularized empirical risk function described in Eq. (10.11) is minimized. Here as well, the model space is extended by allowing third order polynomials. The complexity parameter λ is tuned using a tuning set (which is a subset of the data used to train the model).
- Minimizing ℓ_{a}: the regularized empirical risk given in Eq. (10.12) is minimized. Here as well, the model space is extended by allowing third order polynomials. The complexity parameter λ is tuned using a tuning set (which is a subset of the data used to train the model). Moreover, the orthonormal basis E and corresponding matrix $\mathbf{\Phi}$ that are used to define the ilr-transform consists of the following vectors[4]:

$$\mathbf{\Phi}_{.,1} = \begin{pmatrix} -\sqrt{2} \\ \sqrt{2} \\ 0 \end{pmatrix}, \quad \text{and} \quad \mathbf{\Phi}_{.,2} = \begin{pmatrix} -\sqrt{6} \\ -\sqrt{6} \\ 2\sqrt{6} \end{pmatrix}.$$

[4] This basis was chosen because it is the result of applying the methodology originally proposed in [7] when introducing the ilr-transform.

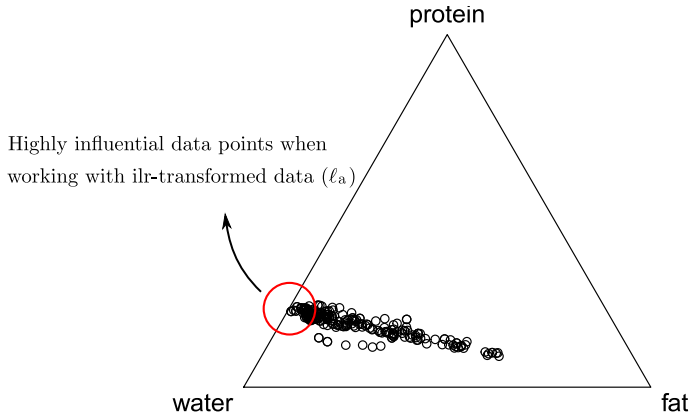


Figure 10.2: The ternary diagram of the *tecator* dataset.

Subsequently, a separate test set was used to estimate the expected value of the empirical loss (for each of the three loss functions). Given a test dataset $T' = ((\mathbf{x}'_i, \mathbf{y}'_i))_{i=1}^m$ and a model $\mathbf{f} : X \to \mathbb{S}^3$ three error measures were computed: (1) the empirical squared loss is computed using $\sqrt{\frac{1}{3m}\sum_{i=1}^m \ell_s(\mathbf{f}(\mathbf{x}'_i), \mathbf{y}'_i)}$ (where the square root was used to enhance interpretability), (2) the empirical deviance is computed using $\frac{1}{3m}\sum_{i=1}^m \ell_d(\mathbf{f}(\mathbf{x}'_i), \mathbf{y}'_i)$ and (3) the loss function based on the Aitchison distance is computed using $\frac{1}{m}\sum_{i=1}^m \ell_a(\mathbf{f}(\mathbf{x}'_i), \mathbf{y}'_i)$.

The *tecator* dataset consists of 240 datapoints. Each datapoint consists of a 3-part composition describing the proportional amounts of fat, meat and water (see Figure 10.2 for an illustration) of a meat sample as well as the NIR spectrum of that sample (100 wavelengths). To investigate the influence of the loss function that is used in the learning process, a subset containing 140 observations was sampled from the original data set and used to train three predictive models (each model was trained with a different loss function). Subsequently, the remaining 100 datapoints were used to evaluate the performance of the model. Each model was evaluated using the error measures defined above. Lastly, to reduce the influence of the particular training dataset that was sampled, this process was repeated 100 times. Table 10.2 reports the arithmetic mean, the standard deviation and the median of the performance measures for each of these runs. Moreover, Table 10.3 reports frequency counts on the number of times (out of 100 runs) that a specific learning strategy outperformed the other strategies for a given performance measure.

When the goal consists of minimizing the performance measure that is based on ℓ_s, it can be seen from Tables 10.2 and 10.3 that the use of ℓ_s in the learning phase outperforms the alternatives. The same conclusion can be drawn in case we want to optimize the measure based on ℓ_d. Here, using ℓ_d in the learning phase outperforms the other methods. It should be noted that the magnitude of the absolute differences of the performances obtained by the different methods is hard

Table 10.2: Performance results of 100 repetitions of the learning process. The column header shows the loss function that was used for learning, the row header shows the loss function that was used for performance evaluation. The numbers denote : "median (mean, standard deviation)" of the performances obtained for 100 runs. The numbers in bold indicate the optimal strategy for each of the performance evaluation measures.

		Training		
		ℓ_a	ℓ_s	ℓ_d
Testing	ℓ_a	0.359 (0.622, 0.663)	1.548 (1.658, 0.913)	**0.281** (0.450, 0.508)
	ℓ_s	0.092 (0.092, 0.025)	**0.052** (0.053, 0.015)	0.082 (0.081, 0.023)
	ℓ_d	0.306 (0.312, 0.027)	0.307 (0.310, 0.020)	**0.297** (0.303, 0.023)

Table 10.3: Performance results of 100 repetitions of the learning process. The column header shows the loss function that was used for learning, the row header shows the loss function that was used for performance evaluation. For example, the number 4 in the second row of the first column shows that, when ℓ_s is used as a performance measure, in 4 out of 100 repetitions training with ℓ_a outperformed the remaining strategies.

		Training		
		ℓ_a	ℓ_s	ℓ_d
Testing	ℓ_a	25	5	70
	ℓ_s	4	95	1
	ℓ_d	11	27	62

to interpret. Moreover, due to the skewness of the histograms of the performances that were obtained, the mean and standard deviation are not very informative. Nevertheless, when comparing the performance results using the frequency counts in Table 10.3, it can be concluded that in 95 out of 100 runs, the approach that minimizes ℓ_s outperformed the other methods when the evaluation measure was ℓ_s. Similarly, in 62 out of 100 runs the approach that minimizes ℓ_d outperformed the other methods when the evaluation measure was ℓ_d.

When the goal exists of minimizing the performance measure that is based on ℓ_a, it can be seen from Tables 10.2 and 10.3 that the model that is trained by minimizing the empirical risk based on ℓ_d will probably outperform a model that is trained by minimizing the empirical risk based on ℓ_a. This may seem somewhat counter-intuitive. However, as indicated in Figure 10.2 multiple datapoints in the *tecator* dataset are located near the boundary of the ternary diagram (in these cases the relative amount of fat is very low). Unfortunately, the ilr-transformation (and the Aitchison geometry in general) is very sensitive to slight variations in those datapoints. Indeed, when transforming the data using the ilr-transform, the ratios that are used lead to coordinates that are strongly influenced by small perturbations of the original dataset, often leading to extreme values. Therefore, noise or rounding errors in those datapoints strongly influence the learning process. On the other hand, it was argued that ℓ_d is quite similar to ℓ_a but can be expected

to be less sensitive to these extreme points. This can explain the fact that, even though the performance is evaluated using ℓ_{a}, a strategy that is based on ℓ_{d} can outperform a strategy that is based on ℓ_{a}.

10.5. Conclusions and discussion

In this chapter, predictive modeling of compositional data was formally introduced and situated within the field of machine learning. Moreover, we briefly introduced the main principles of statistical learning theory (we will be needing some of these principles in the following chapter). Three loss functions were described that can be used when learning to predict compositional data and we highlighted the main differences between these loss functions. Combining the reasoning in this chapter with the information in the introductory chapter on compositional data analysis, we conclude that ℓ_{s} and ℓ_{a} represent two ways of looking at compositional data. While ℓ_{s} focuses on the absolute values of the compositional vector, ℓ_{a} will result in a predictive model that puts emphasis on the relative information in the data. With respect to this dichotomy, ℓ_{d} is hard to situate. Nevertheless, it is a valid loss function that shows some similarity to ℓ_{a}. Moreover, we argued that ℓ_{d} could be used as a robust approximation of ℓ_{a}.

In the experiments with the *tecator* dataset, the distinction between ℓ_{s} and ℓ_{a} reappears. In settings where ℓ_{a} is used to judge the performance of a model, it seems that a model that is learned using ℓ_{s} performs badly. On the other hand, when ℓ_{s} is used to judge the performance of a model, it seems that models that are learned with ℓ_{a} lead to inferior performances. Moreover, the presumption that ℓ_{d} can be used as a robust approximation of ℓ_{a} seems to be confirmed by these experiments. In several cases where ℓ_{a} was used to measure the performance of a model, the model that is learned using ℓ_{d} outperforms the model that is learned using ℓ_{a}.

Some of the findings in this chapter form the basis for the following chapters. In Chapter 11, we focus on the construction of predictive models that minimize an empirical loss function that is based on ℓ_{a}. Due to the close relation between ℓ_{a} and ℓ_{d}, we focus in Chapter 12 on models that minimize an empirical loss function that is based on ℓ_{d}. In both chapters, we will be relying on existing principles and introduce novel ideas that aim to improve the predictive performance of the "traditional" models that have been used in this chapter.

11 Incorporating prior knowledge in multiple-output regression with kernel-based vector functions

11.1. Introduction

The isometric log-ratio transformation (see Chapter 2) is an interesting tool for the predictive modeling of compositional data. This transformation allows compositional data to be transformed to an Euclidean (coordinate) space. Therefore, when the initial output space is $\mathbb{S}_0^q$ with the Aitchison geometry, the log-ratio transformation can be used to map the data into an isometric Euclidean space $\mathbb{R}^{q-1}$. As a result, the initial predictive modeling problem is transformed into an equivalent problem in which we want to learn a function $\mathbf{f} : X \to \mathbb{R}^{q-1}$. This problem is generally known as a multivariate regression problem[1]. Moreover, as opposed to the predictive modeling of compositional outputs, the modeling of multivariate (Euclidean) outputs has been studied for some time (the first papers appearing around 1970). Therefore, we can rely on some well-established methodologies from that field. Nevertheless, it should be mentioned that compared to the univariate output setting, the field is much less mature. Moreover, for many researchers outside the field of predictive modeling these approaches are mostly unknown.

This chapter combines several ideas that can be used to learn high-quality predictive models for vector-valued outputs. More precisely, to obtain these models, we attempt to incorporate prior knowledge or domain knowledge into the learning process. We will mainly rely on the framework of [125] for learning vector-valued functions, introduced in 2005.

The remainder of this chapter is organized as follows:

- In Section 11.2, several approaches to multivariate regression are briefly reviewed.
- In Section 11.3, several types of prior knowledge in the multivariate regression setting are discussed.
- In Section 11.4, an approach to the inclusion of prior knowledge is discussed.
- In Section 11.5, several computational aspects are considered.

[1] Here multivariate refers to the fact that the output space exists of (multivariate) vectors as opposed to traditional (univariate) scalar outputs.

- In Section 11.6, numerical experiments are presented that illustrate the potential of our approach.
- In Section 11.7, conclusions are presented and discussed.

11.2. Multivariate regression approaches

This section contains some background on multivariate regression, as well as several personal insights.

11.2.1. Notation and ridge regression

X will denote the input space, $Y = \mathbb{R}^q$ will denote the q-dimensional Euclidean output space and $\rho_{X,Y}$ will denote a joint distribution over the input-output space. Unless stated differently, we will assume that $X = \mathbb{R}^p$. Similar to many existing studies, we focus on learning a model that obtains optimal predictive performance in terms of label-wise mean squared error. This implies that one intends to find a q-dimensional vector function $\mathbf{f}^*$ that minimizes the expected loss

$$\mathbf{f}^* = \arg\min_{\mathbf{f}\in H} \int_{X\times Y} \sum_{j=1}^{q} \left(y_j - f_j(\mathbf{x})\right)^2 \rho_{X,Y}(\mathbf{x},\mathbf{y}) \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y}\,,$$

over a hypothesis space H of vector functions $\mathbf{f}$ with $f_j : X \to \mathbb{R}$ the j-th component of the vector function and $\mathbf{y} = (y_1, ..., y_q)^\top$. In practice $\rho_{X,Y}$ is unknown and an approximation of $\mathbf{f}^*$ should be learned from an i.i.d. dataset $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\} \subset X \times Y$ (all vectors are column vectors by convention). Furthermore, we write $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)^\top$ and $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^\top$, if X is a vector space. Moreover, we will assume that $\mathbf{X}$ and $\mathbf{Y}$ are centered, *i.e.* the column-wise arithmetic mean is zero. For a generic matrix $\mathbf{A}$, $\mathrm{Vec}(\mathbf{A})$ denotes the vectorization of $\mathbf{A}$ formed by stacking the columns of $\mathbf{A}$ into a single column vector. Additionally, we denote $\overline{\mathrm{Vec}}(\mathbf{A}) = \mathrm{Vec}(A^\top)$. Lastly, given two matrices $\mathbf{A}$ and $\mathbf{B}$, we use $\mathbf{A} \otimes \mathbf{B}$ to denote the Kronecker product.

Now let $q = 1$ (there is only one output). Let the hypothesis space be the space of linear functions, with parameter vector $\mathbf{a}$ (the constant term is zero due to the centering). Using squared loss and L_2-regularization, we obtain the following optimization problem:

$$\underset{\mathbf{a}\in\mathbb{R}^p}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n}(\mathbf{a}^\top \mathbf{x}_i - y_i)^2 + \lambda \left\|\mathbf{a}\right\|_2^2 \,. \tag{11.1}$$

The optimal point $\hat{\mathbf{a}}$ of this optimization problem is the well-known *univariate*

ridge estimator of $\mathbf{a}$ [120]. Matrix notation can be used to write down the closed form solution of this optimization problem:

$$\hat{\mathbf{a}} = (\mathbf{X}^\top \mathbf{X} + n\,\lambda\,\mathbf{I}_p)^{-1}\mathbf{X}^\top \mathbf{Y}\,. \tag{11.2}$$

Notably, when $\lambda = 0$, the resulting estimate is called the ordinary least squares (OLS) estimate.

11.2.2. An example of multivariate regression

In this section, several popular multivariate regression approaches are described. However, let us first consider the following simple example of a multivariate regression setting. Assume that $\mathcal{X}$ is a 2×1 random vector. Moreover, let $\mathbf{a}_1^* \in \mathbb{R}^2$ be a fixed parameter vector such that the 2×1 random vector $\mathcal{Y}$ is[2]

$$\begin{pmatrix} \mathcal{Y}_1 \\ \mathcal{Y}_2 \end{pmatrix} = \underbrace{\begin{pmatrix} a_{1,1}^* & a_{1,2}^* \\ a_{1,1}^* & a_{1,2}^* \end{pmatrix}}_{\mathbf{A}^*} \begin{pmatrix} \mathcal{X}_1 \\ \mathcal{X}_2 \end{pmatrix} + \begin{pmatrix} \mathcal{E}_1 \\ \mathcal{E}_2 \end{pmatrix}$$

where $\mathcal{E}_1$ and $\mathcal{E}_2$ are independent and identically distributed random variables. In short, we have that $\mathcal{Y} = \mathbf{f}^*(\mathcal{X}) + \mathcal{E}$, where $\mathbf{f}^*(\mathbf{x}) = \mathbf{A}^*\mathbf{x}$. Moreover, assume that we have n i.i.d. observations $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ of $\rho_{\mathcal{X},\mathcal{Y}}$ at our disposal. We could fit two separate linear regression models for the first and the second output. For example, for the first output we can use the ridge estimate $\hat{\mathbf{a}} = (\mathbf{X}^\top \mathbf{X} + n\,\lambda\,\mathbf{I}_p)^{-1}\mathbf{X}^\top \mathbf{Y}_{.,1}$. However, as the models for both outputs are identical, we could interpret the observations of $\mathcal{Y}_2$ as additional observations for $\mathcal{Y}_1$. Indeed, this would allow us to generate a dataset containing $2n$ instances. Naturally, as the size of the dataset is doubled, the accuracy of the ridge estimate will improve as well.

This simple example illustrates that the quality of the predictive function that is learned can be improved by taking the relationships between the outputs (or the models corresponding to those outputs) into account. This simple idea is at the heart of most multivariate regression approaches.

11.2.3. Naive multivariate regression

As an initial attempt to solve problems such as the one presented above, a linear vector function $\hat{\mathbf{f}}(\mathbf{x}) = \mathbf{A}\mathbf{x}$ could be used to estimate $\mathbf{f}^*$ where $\mathbf{A}$ is a $q \times p$ parameter matrix. The multivariate ridge estimator $\hat{\mathbf{A}}$ of $\mathbf{A}$ is the solution of the following optimization problem:

[2] Note that $\mathbf{A}_{1,.}^* = \mathbf{A}_{2,.}^*$.

$$\underset{\mathbf{A}\in\mathbb{R}^{q\times p}}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{A}\mathbf{x}_i-\mathbf{y}_i\|_2^2+\lambda\sum_{j=1}^{q}\|\mathbf{A}_{j,.}\|_2^2$$

This optimization problem has the following closed form solution: $\hat{\mathbf{A}}^\top = (\mathbf{X}^\top\mathbf{X} + n\,\lambda\,\mathbf{I}_p)^{-1}\mathbf{X}^\top\mathbf{Y}$. Unfortunately, it can be shown (see for instance [118]) that each of the columns of $\hat{\mathbf{A}}^\top$ can be computed using only the information contained in $\mathbf{X}$ and the corresponding column of $\mathbf{Y}$. Therefore, this approach does not exploit potential relationships or dependencies between the outputs. As a result, we call it a naive multivariate regression approach.

11.2.4. The nature of multivariate regression

In literature, there exist several methods that try to take relationships between outputs into account. Most of these methods use a form of regularization to exploit dependencies between the outputs. Recall from the previous chapter that L_2-regularization forces the entries of $\hat{\mathbf{A}}$ to be small. Therefore, L_2-regularization can be seen as a way to encode our prior knowledge that the true model $\mathbf{f}^*$ can be approximated well by a linear model of which the coefficients are close to zero. Now, let us return to the illustration from the previous section. To encode that the first and the second row of $\mathbf{A}^*$ are highly similar, a suited complexity criterion could be used. For example, the squared norm $\|\mathbf{A}_{1,.} - \mathbf{A}_{2,.}\|_2^2$ could be used to measure the complexity of $\mathbf{A}$. Such a regularization term would encode our knowledge that the true model can be approximated well by a model that has highly similar coefficients for both outputs. Naturally, the information that these rows are (approximately) equal is typically not given in practice. However, this information is partly contained within the sample covariance matrix of $\mathbf{Y}$. Therefore, dataset-specific complexity criteria can be derived from that matrix. Most multivariate regression methods can be shown to fit within the general framework described here.

Below, a procedure is described that is rather frequently used to solve multivariate regression problems. Interestingly, most variants of this method can be shown to fit withing the regularization framework described before. We will be using a similar procedure later in this chapter.

1. Construct a $q \times s$ transformation matrix $\mathbf{U}$ (typically $s \leq q$) such that the columns of $\tilde{\mathbf{Y}} = \mathbf{Y}\mathbf{U}$ are (approximately) uncorrelated and preferable can be predicted easily using the data in $\mathbf{X}$ (we call the resulting variables latent variables).

2. Build s univariate regression models to predict the s latent variables independently. Let $\tilde{\mathbf{f}}$ be the resulting vector function.

3. In the prediction phase, given a new observation $\mathbf{x}$, first compute $\tilde{\mathbf{y}} = \tilde{\mathbf{f}}(\mathbf{x})$. Secondly, use the inverse of the transformation applied in the first step to obtain $\hat{\mathbf{y}}$

11.2.5. Literature on multivariate regression

In this section, we review several multiple output regression methods. Multiple output regression has been extensively studied in statistics, where (often) an underlying linear statistical model is assumed:

$$\mathcal{Y} = \mathbf{A}\mathcal{X} + \mathcal{E}\,, \tag{11.3}$$

with $\mathbf{x}$ (resp. $\mathbf{y}$) a $p \times 1$ (resp. $q \times 1$) real-valued random vector, $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_q)^\top$ a (fixed) $q \times p$ parameter matrix and $\mathcal{E}$ a $q \times 1$ random noise vector. The learning procedure then results in an estimate $\hat{\mathbf{A}}$ of $\mathbf{A}$.

A suboptimal way of handling this problem consists of neglecting the multivariate nature of $\mathcal{Y}$ and learn q independent models by optimizing the empirical least squares on the training data $(\mathbf{X}, \mathbf{Y})$. This gives the OLS estimate $\tilde{\mathbf{A}} = (\tilde{\mathbf{a}}_1, \ldots, \tilde{\mathbf{a}}_q)^\top$. To dominate the OLS estimate in terms of predictive power, most multivariate regression methods provide an estimate $\hat{\mathbf{A}}$ by regularizing $\tilde{\mathbf{A}}$ – see e.g. [126] for a detailed discussion. Examples of such methods are reduced-rank regression [127], FICYREG [128] and the Curds & Whey procedure [126], where $\hat{\mathbf{A}}$ is typically a low-rank approximation of $\tilde{\mathbf{A}}$. As an explanation for the increased predictive performance of such methods, often a connection with James-Stein estimation is put forward. Other methods, such as 2-block partial least squares [129], latent variable methods [130] and the use of weight sharing with neural networks can be enlisted here as well. Even though their behavior has been less analyzed theoretically, they tend to imply a regularizer that exploits the multivariate nature of the data.

Apart from an obvious extension from linear to nonlinear models, modern machine learning algorithms often adopt very similar mechanisms to outperform the baseline of learning a model for every output independently. One of these methods is stacking, a generic approach that has been mainly applied to the related setting of multi-label classification – see e.g. [131, 132]. This method adopts a two-phase procedure. Independent regression models are fitted in a first phase for every output, and the resulting predictions are taken as features in a second regression phase, thereby exploiting potential dependencies between outputs. Notably, the Curds & Whey procedure can be seen as a special case of stacking.

11.2.6. Kernel-based vector-valued functions

Kernel-based scalar functions

In the previous section the hypothesis space was often chosen to be a space of affine functions of the (real-valued) inputs. This situation is rather limiting for at least two reasons. Firstly, there is a clear restriction on the type of functions that can be learned (*i.e.* only affine functions). Secondly, the input space is required to be a real coordinate space. Kernel-based approaches (see for instance [117]) can deal with both limitations. Roughly speaking, kernel methods provide a flexible way of constructing hypothesis spaces. As these spaces are equipped with an inner product these hypothesis spaces are generally called Hilbert spaces. We now elaborate on a specific type of Hilbert spaces that will be used extensively hereafter, *i.e.* reproducing kernel Hilbert spaces. As a starting point, we define a positive definite kernel

Definition 11.1 (Positive Definite Kernel (adapted from [117]))**.** *Let X be a nonempty set. A symmetric function*

$$\begin{aligned} \kappa : X \times X &\to \mathbb{R} \\ (\mathbf{x}, \mathbf{x}') &\mapsto \kappa(\mathbf{x}, \mathbf{x}') \end{aligned}$$

is a positive definite kernel if for all $m \in \mathbb{N}$ and all $\mathbf{x}_1, \ldots, \mathbf{x}_m \in X$, we have that the $m \times m$ matrix $\mathbf{K}$, where $\mathbf{K}_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, is positive definite.

Now, let $k : X \times X \to \mathbb{R}$ be a positive definite kernel. This kernel can be used to define a Hilbert space H of functions. Firstly, given arbitrary $m \in \mathbb{N}$, $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$ and $\mathbf{x}_1, \ldots, \mathbf{x}_m \in X$, we take linear combinations of the form

$$f(.) = \sum_{i=1}^{m} \alpha_i \, \kappa(., \mathbf{x}_i) \, .$$

The set H contains all functions of this form (*i.e.* for arbitrary m, α_i and $\mathbf{x}_i$). Additionally, given

$$f(.) = \sum_{i=1}^{m} \alpha_i \, \kappa(., \mathbf{x}_i) \, , \qquad \text{and} \qquad g(.) = \sum_{j=1}^{n} \beta_j \, \kappa(., \mathbf{x}'_j) \, ,$$

the following bilinear form can be used as inner product:

$$\langle f, g \rangle_H = \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_i \beta_j \kappa(\mathbf{x}_i, \mathbf{x}'_j) \, .$$

Moreover, we have that

$$\langle f, f \rangle_H = \|f\|_H^2 = \sum_{i,j=1}^{m} \alpha_i \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \,.$$

Lastly, it can be shown (see for instance [117]) that $\langle ., . \rangle_H$ is an inner product. Moreover, we have that

$$\langle \kappa(., \mathbf{x}), \kappa(., \mathbf{x}') \rangle_H = \kappa(\mathbf{x}, \mathbf{x}') \,, \tag{11.4}$$

which is called the reproducing property of the kernel. Due to that, H is often called a reproducing kernel Hilbert space.

Even though the derivation above is interesting from a theoretical point of view, the elements of H may be infinite sums. As a result, it is hard to derive an optimization procedure that can be used to search in that type of hypothesis space. However, in a lot of cases, the element of H that optimizes a regularized empirical loss function can be written as a (finite) weighted sum of kernel function evaluations in the training points. The theorem stating this property is known as the representer theorem [117, 133].

Theorem 11.1. *Let H be a kernel Hilbert space with positive definite kernel κ, ℓ a loss function and $\Omega : \mathbb{R} \to \mathbb{R}$ a strictly increasing function, we have that for any given dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n \in (X \times \mathbb{R})^n$ the function $f \in H$ that minimizes*

$$\sum_{i=1}^{n} \ell(f(\mathbf{x}_i), \mathbf{y}_i) + \lambda\, \Omega(\|f\|_H) \,,$$

admits a representation of the form

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i) \,,$$

where $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$.

Kernel-based vector functions

The theory of reproducing kernel Hilbert spaces (RKHSs) for vector-valued functions [125] extends the well-known scalar RKHSs. Most concepts of scalar RKHSs have close parallels in the vector-valued case[1]. The most important difference is the fact that the kernel function is matrix-valued, i.e. $\mathbf{K} : X \times X \to \mathbb{R}^{q \times q}$, where $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)$ is a positive semi-definite matrix for any $\mathbf{x}_i, \mathbf{x}_j \in X$. To stress

[1] We provide only some necessary concepts. For a recent more detailed introduction to the topic, see for instance [134, 135] and the references therein.

the similarity with the scalar case, $\mathbf{K}$ can be described by a scalar kernel $g : (X \times \{1, \ldots, q\})^2 \to \mathbb{R}$ that acts jointly on the objects $\mathbf{x}_i$ and $\mathbf{x}_j$ and the output indices r and $s \in \{1, \ldots, q\}$ [136, 137],

$$(\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j))_{r,s} = g((\mathbf{x}_i, r), (\mathbf{x}_j, s)) \,.$$

Recall that this type of joint input-output kernel differs from the traditional joint kernels encountered in structured output prediction, because only the output indices are used in contrast to considering the full output space. A vector valued RKHS implied by a kernel $\mathbf{K}$ is a space H of vector functions $\mathbf{f} : X \to \mathbb{R}^q$ that can be written as

$$\mathbf{f}(.) = \sum_{i=1}^{\infty} \mathbf{K}(\mathbf{x}'_i, \cdot)\mathbf{c}_i, \qquad \mathbf{x}'_i \in X \,, \mathbf{c}_i \in \mathbb{R}^q \,,$$

where the inner product $\langle \cdot, \cdot \rangle_H$ in H has the reproducing property.

For a given dataset, the empirical L_2-regularized squared error can be written as

$$r_{\mathrm{E}}(\mathbf{f}) = \sum_{j=1}^{q} \sum_{i=1}^{n} \frac{1}{n} (f_j(\mathbf{x}_i) - \mathbf{y}_{i,j})^2 + \lambda \, \|\mathbf{f}\|_H^2 \,. \tag{11.5}$$

The minimizer of r_{E} with respect to $\mathbf{f}$ is given by (an extension of) the representer theorem [137]

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{n} \mathbf{K}(\mathbf{x}_i, \mathbf{x})\mathbf{c}_i \,, \tag{11.6}$$

where $\mathbf{c}_i \in \mathbb{R}^q$ are obtained as

$$\mathbf{c}^* = (\mathbf{K}(\mathbf{X}, \mathbf{X}) + n \, \lambda \mathbf{I}_{qn})^{-1} \mathbf{y}^* \,. \tag{11.7}$$

Here, $\mathbf{c}^* = \overline{\mathrm{Vec}}((\mathbf{c}_1, \ldots, \mathbf{c}_n))$ is an nq vector, moreover $\mathbf{y}^* = \mathrm{Vec}(\mathbf{Y})$ and $\mathbf{K}(\mathbf{X}, \mathbf{X})$ is a block (Gram) matrix

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \begin{bmatrix} \mathbf{K}^{1,1}(\mathbf{X}, \mathbf{X}) & \cdots & \mathbf{K}^{1,q}(\mathbf{X}, \mathbf{X}) \\ \vdots & & \vdots \\ \mathbf{K}^{q,1}(\mathbf{X}, \mathbf{X}) & \cdots & \mathbf{K}^{q,q}(\mathbf{X}, \mathbf{X}) \end{bmatrix}$$

where block $\mathbf{K}^{r,s}(\mathbf{X}, \mathbf{X})$ is an $n \times n$ (scalar) Gram matrix

$$(\mathbf{K}^{r,s}(\mathbf{X}, \mathbf{X}))_{i,j} = g((\mathbf{x}_i, r), (\mathbf{x}_j, s)) \,,$$

where g is defined as before.

Eq. (11.6) is generally called the dual solution of the learning problem. The vectors $\mathbf{c}_i$ are called the dual parameter vectors.

11.3. Prior knowledge for multivariate regression

In this section, we further specify the three types of prior knowledge. 'Prior knowledge' is a very broad term. As such, we start by specifying what we mean with the three types of prior knowledge described before.

Output-output relations. We can consider a setting where the output space has the form $Y = Y_1 \times \ldots \times Y_q$ and prior knowledge allows to map each output Y_j, $j = 1, \ldots, q$, to a (meta-)feature space S with inner product $\langle \cdot, \cdot \rangle_S$. This inner product can be used to describe the similarity between outputs according to this embedding. For a set of q outputs, let $\mathbf{s}_1, \ldots, \mathbf{s}_q \in S$ denote the representation of these outputs in S. The inner products (or similarities) are summarized in a $q \times q$ positive definite matrix $\mathbf{S}$ where

$$\mathbf{S}_{k,l} = \langle \mathbf{s}_k, \mathbf{s}_l \rangle_S \ .$$

As an example, in multiple output regression problems where a set of meta-features describing the outputs is available, the inner product between these features can be used to construct $\mathbf{S}$. However, $\mathbf{S}$ can be more general than this. Indeed, any valid kernel that provides a mapping $S \times S \to \mathbb{R}$ can be used here. For instance, when the relatedness between outputs can be described by a graph[3]. Let us also remark that the relatedness of a pair of outputs is often described through the covariance between several observed values for these outputs. In terms of prior knowledge, $\mathbf{S}$ can be seen as (an approximation of) the population-level covariance between outputs. Indeed, often such a covariance matrix can be considered to describe useful dependencies between outputs. However, as we will show in the experimental section, thoughtless use of such a covariance matrix can deteriorate the predictive performance.

Input-output relations. Information about which features are likely to play a key role for predicting certain outputs can be very helpful to steer the learning process. In this chapter, we will use output-dependent transformations of the input space to represent prior knowledge on input-output relations. Since we will be dealing with kernel methods, this leads in the dual form to the use of an output-dependent kernel. As such, we will define a set of kernels $\kappa_1, \ldots, \kappa_q : X \times X \to \mathbb{R}$.

[3] For example, consider the case where the outputs are the concentrations of a set of metabolites. A metabolic network can be used to derive relationships between these metabolites. More precisely, the metabolites can be represented as the nodes of a graph and the presence of a relationship between two metabolites can be encoded by means of an edge. The resulting graph can be used to measure the overall relatedness of the metabolites and define a mapping of the metabolites into a (meta-)feature space. The diffusion kernel [138] implies such a mapping and can be used to encode the domain knowledge (the information from the metabolic network) into the learning problem.

For each output, the learning scheme should allow the corresponding kernel to dominate the learning process. In addition, output-independent kernels can be used, as if no prior knowledge was available.

Input-input relations. In the spirit of [139], we can consider a setting where the feature space has the form $X = X_1 \times \ldots \times X_p$ and prior knowledge allows to map each feature X_j, $j = 1, \ldots, p$, to a meta-feature space M with inner product $\langle \cdot, \cdot \rangle_M$. Let $\mathbf{m}_k$ and $\mathbf{m}_l \in M$ be the representations of X_k and X_l, then the similarity between X_k and X_l can be expressed as $\langle \mathbf{m}_k, \mathbf{m}_l \rangle_M$. Combining all pairwise inner products results in a positive definite matrix $\mathbf{M}$ with entries $\mathbf{M}_{k,l} = \langle \mathbf{m}_k, \mathbf{m}_l \rangle_M$. In practice, prior knowledge can be present explicitly by means of meta-features. More general, a kernel function that maps pairs of features to a real value can be used as well. When $X = \mathbb{R}^p$ one might use the sample covariance matrix (or a smoothened version thereof) for M. This would mean that the embedding of a feature is obtained by observing its value for several test points. When the Mahalanobis distance is used as a distance measure on the input space, this is exactly what is happening. Once more, thoughtless use of such a covariance matrix can deteriorate the predictive performance.

11.4. Including prior knowledge in kernel-based vector-valued functions

We will use the kernel function $\mathbf{K}$ to include the three types of prior knowledge defined before. In what follows, we will see that all three types of prior knowledge can be encoded into a kernel of the form

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l=1}^{m} \kappa(\mathbf{x}_i, \mathbf{x}_j) \mathbf{B}_l \,, \tag{11.8}$$

where $\kappa : X \times X \to \mathbb{R}$ is a scalar kernel and $\mathbf{B}_l$ is a positive definite matrix as defined below. In what follows, we describe how the three types of prior knowledge can be incorporated in such a kernel.

11.4.1. Input-input knowledge

As this type of prior knowledge is present in single output-learning as well, we can use results from the extensive literature on prior-knowledge incorporation for single outputs. The prior knowledge that can be available is often very application-specific and we will provide several examples in the experimental section. However, for now assume that this prior knowledge leads to a scalar kernel $\kappa : X \times X \to \mathbb{R}$.

11.4.2. Output-output knowledge

The separable kernel[4]

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{B}\,, \tag{11.9}$$

where $\mathbf{B}$ is a (fixed) positive semi-definite matrix is a simple construction to transform a scalar kernel into a matrix-valued kernel. It turns out that a lot of literature about matrix-valued kernels leads to kernels of this type (see [134] for an extensive overview). It is clear that, for this kind of kernel, the dependencies between the outputs are contained within $\mathbf{B}$. As prior knowledge on output-output relations allows to define a positive definite matrix $\mathbf{S}$ (see Section 11.3) that provides a rough estimate of these dependencies, we may set $\mathbf{B} = \mathbf{S}$. However, even though this approach leads to a valid kernel, it remains to be seen whether the resulting kernel effectively captures our prior knowledge. In the remainder of this paragraph, we present several results (partly theoretical, partly based on simulation experiments), that can be used to gain insight into this problem. The goal of the discussion presented hereafter is to compare the quality of several strategies that can be adopted to choose $\mathbf{B}$.

We assume the following multivariate (statistical) model :

$$\begin{aligned} \mathcal{Y} &= \mathbf{A}^* \mathcal{X} + \mathcal{E} \\ &= \mathbf{f}^*(\mathcal{X}) + \mathcal{E}\,, \end{aligned} \tag{11.10}$$

where $\mathcal{X}$ is a p-dimensional normally distributed random vector with $\mathbb{E}[\mathcal{X}] = \mathbf{0}_p$ and $\mathrm{cov}(\mathcal{X}) = \Sigma$ and $\mathcal{E}$ is normally distributed with $\mathbb{E}[\mathcal{E}] = \mathbf{0}_q$ and $\mathrm{cov}(\mathcal{E}) = \sigma\,\mathbf{I}_q$. $\mathbf{A}^*$ is a (fixed) $q \times p$ matrix.

For a given dataset T containing n independent observations of $(\mathcal{X}, \mathcal{Y})$, and a hypothesis space H implied by the matrix-valued kernel function $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{B}$, the minimizer of the empirical L_2-regularized squared error defined in Eq. (11.5), *i.e.* $\hat{\mathbf{f}} = \arg\min_{\mathbf{f}\in H} r_E(\mathbf{f})$ is used to estimate $\mathbf{f}^*$. Subsequently, to measure the quality of $\hat{\mathbf{f}}$, the risk is used

$$r(\hat{\mathbf{f}}) = \int_{X\times Y} \sum_{j=1}^{q} \left(y_j - \hat{f}_j(\mathbf{x})\right)^2 \rho_{X,Y}(\mathbf{x}, \mathbf{y})\,\mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{y}\,.$$

Naturally, the estimate $\hat{\mathbf{f}}$ and the risk $r(\hat{\mathbf{f}})$ depend on the training dataset T. However, as T consists of i.i.d. observations of $(\mathcal{X}, \mathcal{Y})$, this dataset can be interpreted as a random variable as well. Therefore, an interesting measure of the quality of an estimation procedure can be obtained by computing the expected value of

[4] Separable kernels are, following [135], matrix-valued kernels that can be written in the form presented in Eq. (11.9). These kernels are called separable as they can be written as a product of a scalar kernel (encoding the relatedness between the inputs) and a matrix (encoding the dependencies between the outputs).

$r(\hat{\mathbf{f}})$, denoted $\mathbb{E}[r(\hat{\mathbf{f}})]$ where the expectation runs over all training datasets of size n.

In this study, we let $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$. In this case, the implied hypothesis space H is a set of linear functions. Computing $\mathbb{E}[r(\hat{\mathbf{f}})]$ in this setting is often referred to as a *random design analysis* (see for instance [140] for an analysis in the univariate case). Unfortunately, this type of analysis is rather complicated. Therefore, we further restrict ourselves to cases where the sample covariance matrix of the sample is identical to the population covariance matrix, *i.e.* $\mathbf{X}^\top \mathbf{X} = n\,\Sigma$. This case is often referred to as a *fixed design analysis*. Moreover, it can be shown that (see for instance [140]) the risk associated with the minimizer of r_E is not affected by a rotation of the input space. Therefore, we can, without loss of generality assume that Σ is a diagonal matrix. The diagonal elements are equal to the eigenvalues of Σ and are denoted $\gamma_1, \ldots, \gamma_p$.

Now, let $\hat{\mathbf{f}}$ be the (random) minimizer of the L_2-regularized risk, with $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j \mathbf{B}$ and given $\lambda > 0$. Moreover, let $\mathbf{UDU}^\top$ be the eigendecomposition of $\mathbf{B}$ such that columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{B}$. In that case, we have that

$$\mathbb{E}[r(\hat{\mathbf{f}}_i)] = \underbrace{\frac{\sigma^2}{n} \sum_{j=1}^{q} \mathbf{U}_{i,j}^2 \sum_{k=1}^{p} \left(\frac{\gamma_k}{\gamma_k + \frac{\lambda}{\mathbf{D}_{j,j}}} \right)^2}_{\text{variance}} + \underbrace{\sum_{k=1}^{p} \gamma_k \left(\sum_{j=1}^{q} \left(\frac{\gamma_k}{\gamma_k + \frac{\lambda}{\mathbf{D}_{j,j}}} - 1 \right) \mathbf{A}_{k,.} \mathbf{U}_{.,j} \mathbf{U}_{i,j} \right)^2}_{\text{bias}} . \quad (11.11)$$

The technical details concerning the derivation of this equation are given in Appendix 11.A. As indicated in this equation, $\mathbb{E}[r(\hat{\mathbf{f}}_i)]$ is decomposed into a part that can be attributed to the bias of the procedure and a part that can be attributed to the variance of the procedure. This formula can be used to investigate the performance of a specific strategy for choosing $\mathbf{B}$. Figures 11.1 and 11.2 visualize several results obtained with this formula. More precisely, to obtain these figures, the statistical model given in Eq. (11.10) was used with:

$$\Sigma = \begin{pmatrix} 4 & 0 \\ 0 & 10 \end{pmatrix}, \qquad \sigma = 1\,, \qquad n = 10\,, \qquad \mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2)^\top\,,$$

$$\mathbf{a}_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \qquad \mathbf{a}_2 = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \mathbf{a}_1\,.$$

The rotation angle θ guides the similarity between the parameter vectors for the

two outputs. It can be seen that, for $\theta = 0$, we have that $\mathbf{a}_1 = \mathbf{a}_2$. On the other hand, for $\theta = \pi/2$, the parameter vectors are orthogonal, whereas for $\theta = \pi$ we have that $\mathbf{a}_1 = -\mathbf{a}_2$. Finally, it must be noted that $\mathbb{E}[r(\hat{\mathbf{f}}_1)]$ is a function of the regularization parameter λ. As Figures 11.1 and 11.2 are used to compare several strategies, each strategy should have its own 'optimal' value for λ (for each θ). Therefore, for every rotation angle and every method, the value of λ that minimizes $\mathbb{E}[r(\hat{\mathbf{f}}_1)]$ was computed (using a bisection search).

The green lines in Figures 11.1 and 11.2 represent the case where $\mathbf{B} = \text{cov}(\mathcal{Y})$. On the other hand the blue lines represent the case where $\mathbf{B} = \text{cov}(\mathcal{Y}) - \text{cov}(\mathcal{E})$ and the red lines represent the case where the models are learned independently. From Figure 11.2 it can be seen that, when both parameter vectors are highly similar, including prior knowledge into the learning process is capable of reducing the expected error $\mathbb{E}[r(\hat{\mathbf{f}}_1)]$. Moreover, at the cost of a slight increase in the bias, these strategies allow for a drastic reduction of the variance as compared to learning both models independently. When the models are very dissimilar (around $\theta = \pi/2$) the use of the covariance matrices does not lead to a decrease in the expected error. Moreover, there is a clear symmetry in the effect of the rotation angle.

Lastly, the black lines represent the situation where $\mathbf{B}$ is not given a priori, but needs to be estimated from data. In this study, we used $\mathbf{B} = \frac{1}{n}\mathbf{Y}^\top\mathbf{Y}$. Unfortunately, we were unable to derive an analytical expression for this error. Therefore, a simulation experiment was set up in which $\mathbf{X}$ and $\mathbf{E}$ (the matrix of error terms) were sampled from bivariate normal distributions (mean and covariance matrices were equal to the matrices used for the other experiments). For each θ, we averaged over 10000 repetitions. From Figures 11.1 and 11.2, it can be seen that estimating $\mathbf{B}$ by means of the sample covariance matrix of the outputs is only beneficial when the parameter vectors of both outputs are highly similar. For all other cases, learning the outputs independently leads to a smaller error. However, it should be noted that, as the number of outputs increases, the benefit from using multivariate regression methods (as compared to univariate variants) may increase as well. For a large number of outputs, the sample-based estimate of $\mathbf{B}$ may outperform the independently learned models in a number of settings. However, we were unable to verify this assertion in a quantitative manner.

11.4.3. Input-output knowledge

Recall the example from Section 11.2, where the models of both outputs were identical. We now consider the opposite case, where the relationship between the

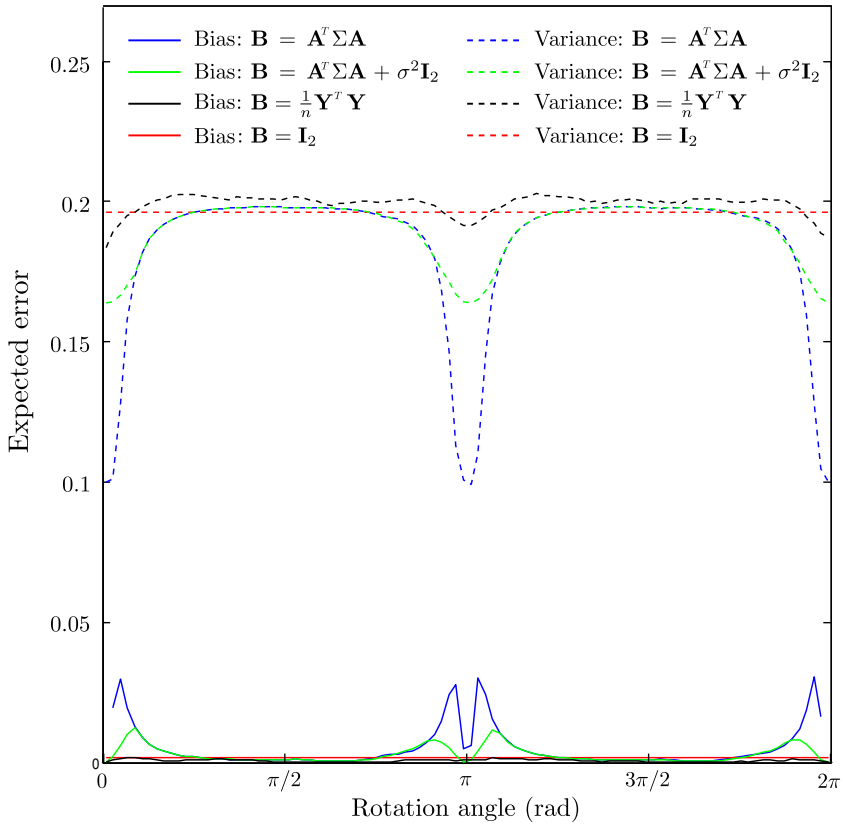


Figure 11.1: The expected error $\mathbb{E}[r(\hat{\mathbf{f}}_1)]$, decomposed in a bias and a variance part, according to different strategies for determining $\mathbf{B}$.

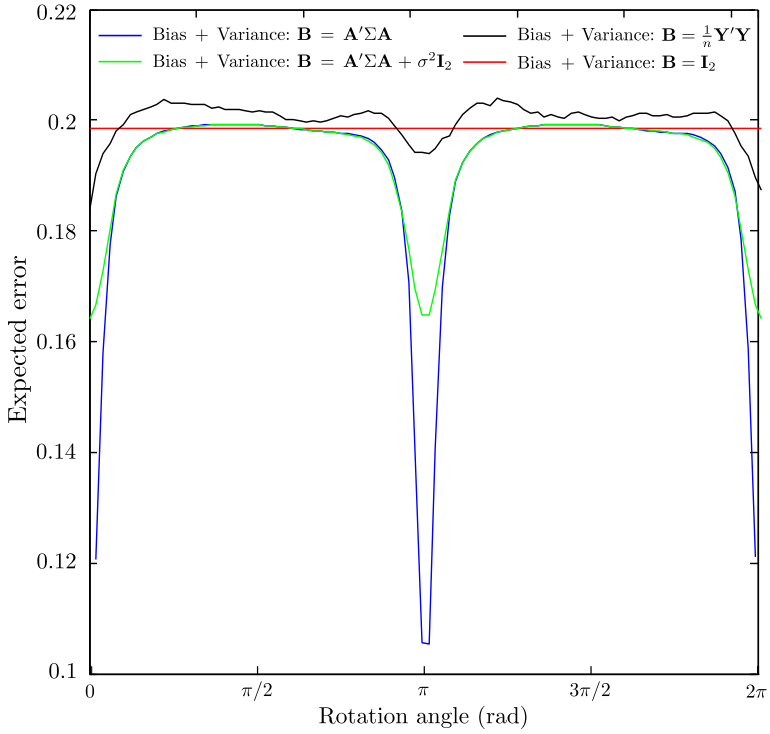


Figure 11.2: The expected error $\mathbb{E}[r(\hat{\mathbf{f}}_1)]$ according to different strategies for determining $\mathbf{B}$.

models is very weak, *i.e.*

$$\begin{pmatrix} \mathcal{Y}_1 \\ \mathcal{Y}_2 \end{pmatrix} = \underbrace{\begin{pmatrix} a^*_{1,1} & 0 \\ 0 & a^*_{2,2} \end{pmatrix}}_{\mathbf{A}^*} \begin{pmatrix} \mathcal{X}_1 \\ \mathcal{X}_2 \end{pmatrix} + \begin{pmatrix} \mathcal{E}_1 \\ \mathcal{E}_2 \end{pmatrix} .$$

The remainder of the experimental setup remains the same.

From this model, it can be seen that $\mathcal{Y}_1$ only depends on the value of the first input and $\mathcal{Y}_2$ only depends on the value of the second input. This type of knowledge can be incorporated into the estimation procedure of $\mathbf{A}^*$ by using a matrix-valued kernel of the following form:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \begin{pmatrix} \kappa_1(\mathbf{x}, \mathbf{x}') & 0 \\ 0 & \kappa_2(\mathbf{x}, \mathbf{x}') \end{pmatrix} ,$$

with $\kappa_1(\mathbf{x}, \mathbf{x}') = x_1\, x'_1$ and $\kappa_2(\mathbf{x}, \mathbf{x}') = x_2\, x'_2$. According to the representer theorem, the minimizer of the L_2-regularized squared loss (Eq. (11.5)) for this kernel can be written as

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{n} \begin{pmatrix} \kappa_1(\mathbf{x}, \mathbf{x}') & 0 \\ 0 & \kappa_2(\mathbf{x}, \mathbf{x}') \end{pmatrix} \begin{pmatrix} c_{i,1} \\ c_{i,2} \end{pmatrix} ,$$

From this construction, it is clear that the model that is learned will respect our prior knowledge. Naturally, this example is rather extreme. Therefore, let us assume a more general case where both models can share information. This can be accomplished by using the following kernel:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = (\kappa_1(\mathbf{x}, \mathbf{x}') + \kappa_2(\mathbf{x}, \mathbf{x}'))\, \mathbf{B} + \begin{pmatrix} \kappa_1(\mathbf{x}, \mathbf{x}') & 0 \\ 0 & \kappa_2(\mathbf{x}, \mathbf{x}') \end{pmatrix} .$$

Notably, this kernel can be rewritten as follows:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \kappa_1(\mathbf{x}, \mathbf{x}') \begin{pmatrix} \mathbf{B}_{1,1} + 1 & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{pmatrix} + \kappa_2(\mathbf{x}, \mathbf{x}') \begin{pmatrix} \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,1} & \mathbf{B}_{2,2} + 1 \end{pmatrix} .$$

We now generalize and parametrize this approach. Assume that, for a multivariate output regression problem consisting of q outputs, prior knowledge of the two types described above leads to kernel κ and matrix $\mathbf{B}$. If we additionally possess of output-specific knowledge expressed through the kernels $\kappa_1, \ldots \kappa_q$, the following

kernel combines all predefined sources of prior knowledge:

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{B} + \alpha \left(\kappa_1(\mathbf{x}_i, \mathbf{x}_j) \begin{bmatrix} \mathbf{B}_{1,1} + \gamma & \cdots & \mathbf{B}_{1,q} \\ \vdots & & \vdots \\ \mathbf{B}_{q,1} & \cdots & \mathbf{B}_{q,q} \end{bmatrix} \right.$$
$$\left. + \ldots + \kappa_q(\mathbf{x}_i, \mathbf{x}_j) \begin{bmatrix} \mathbf{B}_{1,1} & \cdots & \mathbf{B}_{1,q} \\ \vdots & & \vdots \\ \mathbf{B}_{q,1} & \cdots & \mathbf{B}_{q,q} + \gamma \end{bmatrix} \right), \quad (11.12)$$

where α and γ are two parameters. The values of α and γ determine the extent to which output-specific prior knowledge is incorporated into the learning procedure. Firstly, when γ is large compared to $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ and the entries in $\mathbf{B}$, and $\alpha \approx 1$, outputs are learned independently, and each output only uses its output-specific kernel. Secondly, when γ is large compared to $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ and the entries in $\mathbf{B}$ and $\alpha \approx 1/\gamma$, the output-specific kernels are used separately for each output but outputs are related through the first term $\kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{B}$. Thirdly, when γ is in the order of magnitude of the entries in $\mathbf{B}$ and $\alpha \approx 1$, the output-specific nature of $\kappa_1, \ldots, \kappa_q$ is somewhat reduced, and the kernels are shared among outputs. As a special case, when $\gamma = 0$, no output-specificity is used, as (11.12) can be written as $\tilde{k}(\mathbf{x}_i, \mathbf{x}_j)\mathbf{B}$, where $\tilde{k}(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j) + \alpha(\kappa_1(\mathbf{x}_i, \mathbf{x}_j) + \ldots + \kappa_q(\mathbf{x}_i, \mathbf{x}_j))$.

11.5. Computational aspects

11.5.1. Directly computing the dual parameters

Optimizing the L_2-regularized squared error in Eq. (11.5) leads to an optimization problem for which the closed form solution in the dual form is given by Eq. (11.7). However, the closed form solution requires the inverse of $(\mathbf{K}(\mathbf{X}, \mathbf{X}) + n\,\lambda\mathbf{I}_{qn})$ to be computed. Unfortunately, as $\mathbf{K}(\mathbf{X}, \mathbf{X})$ is an $nq \times nq$ matrix, the complexity of computing this inverse is of the order $\mathcal{O}(q^3n^3)$, which is too expensive for many real-life datasets. Furthermore, when $\mathbf{K}(\mathbf{X}, \mathbf{X})$ is non-sparse, the amount of computer memory required to store this matrix ($\mathcal{O}(q^2n^2)$) can become excessive even for medium-sized datasets. Fortunately, by exploiting several mathematical properties of the kernel function that we proposed, more efficient procedures can be derived to compute $\mathbf{c}^*$. For separable kernels of the form $\kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{B}$, Baldassarre *et al.* [141] provide a simple procedure that avoids the explicit computation of $\mathbf{K}(\mathbf{X}, \mathbf{X})$, leading to a complexity $\mathcal{O}(q^3 + n^3)$, making the method applicable in practice. However, their procedure cannot be applied to kernels of the form given in (11.8). As a result, this procedure can not be used to solve learning problems that use (11.8) in a computationally tractable manner. Therefore, in the following

paragraphs we focus on developing a procedure that can be used to fit models that use (11.8).

11.5.2. An efficient conjugate gradient procedure

In what follows, we derive a conjugate gradient method that can be used to obtain an accurate approximation of $\mathbf{c}^*$ in an efficient manner, both in terms of memory requirements and computational complexity, for kernels of the form given by Eq. (11.8).

For the reader who is unfamiliar with conjugate gradient methods, we elaborate on the general principles of these methods hereafter, before applying them to optimize the empirical regularized risk. This description is inspired on [11].

The linear conjugate gradient method

Essentially, the linear conjugate gradient method [142] is a method that is used for solving systems of linear equations:

$$\mathbf{A}\mathbf{x} = \mathbf{b}\,,$$

where $\mathbf{A}$ is an $m \times m$ symmetric positive definite matrix and $\mathbf{b}$ is an $m \times 1$ vector. As a first step, the linear system is transformed into the following equivalent optimization problem (*i.e.* a problem of which the optimal point solves the system):

$$\underset{\mathbf{x}\in\mathbb{R}^m}{\text{minimize}} \quad \frac{1}{2}\mathbf{x}^\top\mathbf{A}\mathbf{x} - \mathbf{b}^\top\mathbf{x}\,.$$

A steepest gradient descent procedure could be used to solve this problem. Due to the convexity of the problem, such an approach will converge to the global optimum. However, the number of iterations that is required may be rather large. Moreover, as the gradient direction can be expensive to compute, the whole approach may be computationally expensive. Conjugate gradient methods use an alternative set of search directions that can be computed easily. Moreover, it can be shown that at most m iterations are required to obtain convergence. Below, we present the pseudo-code of the linear conjugate gradient method.

procedure $\mathrm{CG}(\mathbf{A}, \mathbf{b}, \mathbf{x}_0)$
  $\mathbf{r}_0 \leftarrow \mathbf{A}\mathbf{x}_0 - \mathbf{b}$, $\mathbf{p}_0 \leftarrow -\mathbf{r}_0$, $k \leftarrow 0$
  while $\mathbf{r}_0 \neq \mathbf{0}_m$ **do**

$$
\begin{aligned}
\alpha_k &\leftarrow \frac{\mathbf{r}_k^\top \mathbf{r}_k}{\mathbf{p}_k^\top \mathbf{A}\mathbf{p}_k} \\
\mathbf{x}_{k+1} &\leftarrow \mathbf{x}_k + \alpha_k \mathbf{p}_k \\
\mathbf{r}_{k+1} &\leftarrow \mathbf{r}_k + \alpha_k \mathbf{A}\mathbf{p}_k \\
\beta_{k+1} &\leftarrow \frac{\mathbf{r}_{k+1}^\top \mathbf{r}_{k+1}}{\mathbf{r}_k^\top \mathbf{r}_k} \\
\mathbf{p}_{k+1} &\leftarrow -\mathbf{r}_{k+1} + \beta_{k+1}\mathbf{p}_k \\
k &\leftarrow k+1
\end{aligned}
$$

  end while
end procedure

In this procedure, $\mathbf{r}_k$ is the residual $(\mathbf{A}\mathbf{x}_k - \mathbf{b})$ of the linear system at iteration k. Interestingly, $\mathbf{r}_k$ is also equal to the derivative of the objective at iteration k. Moreover, $\mathbf{p}_k$ and α_k represent the search direction at iteration k and the step length in that direction. The main computational task that needs to be performed here is the computation of $\mathbf{A}\mathbf{p}_k$ (the computations $\mathbf{p}_k^\top(\mathbf{A}\mathbf{p}_k)$ and $\mathbf{r}_k^\top \mathbf{r}_k$ are mostly less demanding). This means that in order to obtain an efficient numerical scheme, we need to be able to compute the matrix-vector product $\mathbf{A}\mathbf{p}_k$ efficiently. This will be the main objective of the remainder of this section.

Optimizing the empirical risk

We start by noting that, for the matrix-valued kernel $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l=1}^m \kappa_l(\mathbf{x}_i, \mathbf{x}_j)\mathbf{B}_l$, $\mathbf{K}(\mathbf{X}, \mathbf{X})$ can be written as

$$
\mathbf{K}(\mathbf{X}, \mathbf{X}) = \sum_{l=1}^m \mathbf{B}_l \otimes \kappa_l(\mathbf{X}, \mathbf{X})\,,
$$

where $\kappa(\mathbf{X}, \mathbf{X})$ denotes the Gram matrix for kernel κ_l and (training) matrix $\mathbf{X}$, *i.e.* $\kappa_l(\mathbf{X}, \mathbf{X})_{i,j} = \kappa_l(\mathbf{x}_i, \mathbf{x}_j)$. Moreover, the normal equations obtained for solving Eq. (11.5) in that case are

$$
\left(\sum_{l=1}^m \mathbf{B}_l \otimes k_l(\mathbf{X}, \mathbf{X}) + n\,\lambda\,\mathbf{I}_{qn}\right)\overline{\mathbf{c}} = \mathbf{y}^*\,, \tag{11.13}
$$

which is a system of linear equations in $\overline{\mathbf{c}}$ with a positive definite coefficient matrix. As such, conjugate gradient methods can be used to generate a sequence of vectors $\overline{\mathbf{c}}$ that converges to $\mathbf{c}^*$. The order of complexity of one iteration of a conjugate gradient method equals the order of complexity of multiplying the coefficient matrix

with a vector. For a vector $\mathbf{a} \in \mathbb{R}^{nq}$, we have that

$$\left(\sum_{l=1}^{m} \mathbf{B}_l \otimes k_l(\mathbf{X},\mathbf{X}) + n\,\lambda\,\mathbf{I}_{qn}\right)\mathbf{a} = \overline{\mathrm{Vec}}\left(\sum_{l=1}^{m} \mathbf{B}_l \mathbf{A} k_l(\mathbf{X},\mathbf{X})\right) + n\,\lambda\mathbf{a}\,, \qquad (11.14)$$

where $\overline{\mathrm{Vec}}(\mathbf{A}) = \mathbf{a}$. The complexity of this multiplication is $\mathcal{O}((q^2n + qn^2)m)$. Moreover, this multiplication avoids the explicit computation of $\mathbf{K}(\mathbf{X},\mathbf{X})$, resulting in strongly reduced memory requirements.

When using the kernel given in Eq. (11.12), we have $m = q + 1$ and a complexity of $\mathcal{O}(q^3n + q^2n^2)$. However, a further reduction is possible by observing that this kernel can be written as the sum of a dense component and a sparse component. Let $\mathbf{O}^l(\gamma)$ denote a $q \times q$ matrix where all entries are zeros except for the l-th diagonal element, which is equal to γ. Using this notation, Eq. (11.12) can be rewritten as

$$\mathbf{K}(\mathbf{x}_i,\mathbf{x}_j) = \left(\kappa(\mathbf{x}_i,\mathbf{x}_j) + \alpha\left(\sum_{l=1}^{q}\kappa_l(\mathbf{x}_i,\mathbf{x}_j)\right)\right)\mathbf{B} + \sum_{l=1}^{q}\alpha\,\kappa_l(\mathbf{x}_i,\mathbf{x}_j)\mathbf{O}^l(\gamma)\,,$$

the resulting kernel (Gram) matrix being

$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{B}\otimes\left(\kappa(\mathbf{X},\mathbf{X}) + \alpha\left(\sum_{l=1}^{q}\kappa_l(\mathbf{X},\mathbf{X})\right)\right) + \sum_{l=1}^{q}\alpha\,\mathbf{O}^l(\gamma)\otimes\kappa_l(\mathbf{X},\mathbf{X})\,.$$

As $\mathbf{O}^l(\gamma)$ contains only one non-zero entry, the computational cost of multiplying a vector with the second term of this kernel matrix can now be performed with a complexity of $\mathcal{O}(n^2q)$ (instead of $\mathcal{O}(n^2q^2)$ for the non-sparse case). This reduces the total computational cost of one iteration to $\mathcal{O}(q^2n + qn^2)$. To solve system (11.13) exactly, at most qn iterations are needed. However, as we will illustrate in the experimental section, the actual number of iterations needed to obtain an acceptable approximation is much smaller. Moreover, it has been noted on several occasions that early-stopping has a beneficial regularizing effect [141].

11.6. Experimental results

In this section, we demonstrate the potential of our approach. As such, the goal of this section is twofold. Firstly, we show that including prior knowledge using the methodology described before can lead to a model with increased predictive power. We will illustrate this on a series of artificial test problems. Secondly, we show that the optimization procedure that is applied is capable of handling high-dimensional datasets, both in terms of the number of inputs as in the number of outputs.

11.6.1. An artificial problem

As an artificial problem, we consider the linear regression model (11.3). In this experiment, dependencies between the outputs were acquired by including dependencies between the rows of $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_q)^\top$ through the following hierarchical model. Firstly, a prototype vector $\mathbf{a}^* \in \mathbb{R}^p$ is drawn from a multivariate normal distribution with mean $\mathbf{0}_p$ and an identity covariance matrix. Using this prototype, $\mathbf{a}_1, \ldots, \mathbf{a}_q$ are constructed as

$$\begin{cases} \Pr(a_{r,i} = a_i^*) &= 0.6 \\ \Pr(a_{r,i} = 0) &= 0.4 \end{cases}$$

where $\Pr(a_{r,i} = a_i^*)$ denotes the probability that the ith element of $\mathbf{a}_r$ equals the ith element of $\mathbf{a}^*$. Moreover, to generate train and test sets, we choose $\mathcal{X} \sim \mathcal{N}(\mathbf{0}_p, \mathbf{I}_p)$ and $\mathcal{E} \sim \mathcal{N}(\mathbf{0}_q, 1.5\,\mathbf{I}_q)$. As it is our aim to illustrate that using prior knowledge can improve the predictive performance, we will optimize the empirical L_2-regularized risk function, using a range of kernels, encoding different types of prior knowledge. As a baseline (referred to as SEP-VAR), outputs are learned independently. To this end, we use a separable kernel (11.9) with $\mathbf{B}$ a diagonal matrix, where the ith diagonal element is the variance of the ith output computed on training data. As a first competitor, we use the separable kernel, where $\mathbf{B}$ is the covariance matrix on training data (SEP-COV-Tr). Subsequently, we use the separable kernel, where $\mathbf{B}$ is the population covariance matrix (SEP-COV-Pop), exploiting knowledge of the population covariance as prior knowledge. As a final alternative (COV-Pop-IO), we use the newly introduced kernel (11.12). The output-specific kernels are chosen as linear kernels on a subset of the inputs, i.e. input $\mathbf{x}_i$ is included in the computation of output-specific kernel κ_r with probability 0.8 if $a_{r,i} \neq 0$ and with probability 0.2 if $a_{r,i} = 0$. For all experiments, we set $q = p = 30$.

Table 11.1 reports, for training sets of variable size, the mean squared prediction error on separate test sets. The reported errors are averaged over 50 replicates of the data generation procedure. The regularization parameter λ was tuned using nested cross-validation; $\alpha = 1/1000$ and $\gamma = 1000$ were both kept fixed. Note that these parameter setting imply high output-specificity of the output-specific kernels. As such, every method had only one parameter to be tuned. From Table 11.1, it can be seen that the inclusion of prior knowledge is capable of boosting the performance of models that are learned by optimizing the empirical L_2-regularized squared loss. It is clear that learning tasks independently leads to inferior models. Moreover, this table suggests a relation between the size of the training set and the effectiveness of including prior knowledge. Indeed, the results presented in this table suggest that the inclusion of prior knowledge is most beneficial in data-scarce settings.

Table 11.1: Mean squared error on artificial data using four alternatives for the kernel function. The last column shows results for the malicious example.

Method	$n = 10$	$n = 15$	$n = 25$	$n = 50$	Malicious
SEP-VAR	145.18	115.15	82.15	38.51	1.34
SEP-COV-Tr	144.67	114.95	72.79	32.87	1.75
SEP-COV-Pop	144.36	112.70	68.29	29.02	1.45
COV-Pop-IO	141.93	107.98	67.61	30.41	-

In addition to these experiments, Table 11.1 shows a simple result when prior knowledge on correlations on the outputs is used wrongly. For this experiment, the goal is to learn the identity map between inputs and outputs, i.e. the ith output equals the ith input with some additive noise ($\mathcal{E} \sim \mathcal{N}(\mathbf{0}^p, 0.5\mathbf{I}_q)$). Correlation between outputs is now introduced by sampling the inputs from a multivariate normal distribution with a covariance matrix $\mathbf{V} \neq \mathbf{I}_p$. As such, even though the models for different outputs are strongly dissimilar, there exists correlation between the outputs. As such, using this correlation in a manner that forces models to become more similar is expected to lead to a decreased performance. For this experiment, we choose $p = q = n = 10$.

11.6.2. Assessing the computational complexity

A similar experimental setup can be used to assess the computational complexity of the conjugate gradient based optimization procedure. As shown in Section 3, the time complexity of one conjugate gradient iteration is $\mathcal{O}(q^2n + n^2q)$. However, in the worst case, the number of iterations needed to converge is np. Figure 11.3 shows that this worst-case complexity is unlikely to occur. The left panel illustrates that the number of iterations that is needed decreases as n increases, seemingly to converge to a constant value, suggesting that (at least in this setting) the actual number of iterations will not become excessive as n increases. on the other hand, the right panel suggests the number of iterations that is required to be a linear function of the number of outputs q.

11.6.3. An illustration in agriculture, with discussion

The fatty acid composition of the milk of lactating cows is a rich source of information. For example, the abundance of several fatty acids can be used as an indicator for the presence of metabolic disorders in the lactating animal (such as for instance ketosis [143]). Additionally, when milk is used for consumption its fatty acid composition might be of interest to the consumer as it may, in turn, affect a consumer's health [144]. For those reasons, in the dairy industry, there exists a general interest in methodologies that can be used to assess the fatty acid

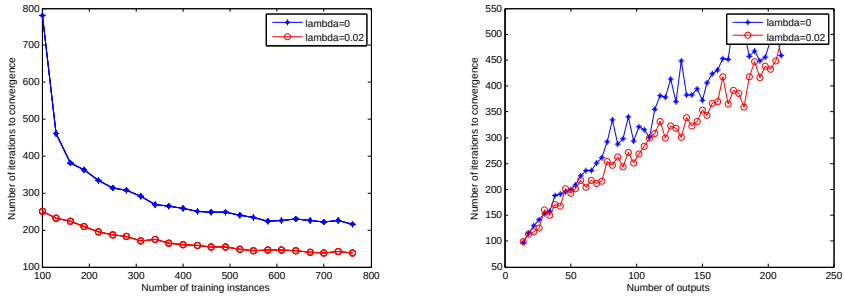


Figure 11.3: Number of iterations needed to achieve convergence: (left) versus number of instances with $q = 30$ fixed, (right) versus number of outputs, with $n = 50$ fixed. For both settings, we set $p = 100$. Two values for the regularization parameter λ were chosen: 0 (no regularization) and 0.02 (an approximately optimal value determined with cross-validation)

composition of milk samples in an accurate and efficient manner. One of these methodologies uses spectrophotometry. More precisely, these methodologies use the near infra red spectrum or the Raman spectrum of a milk sample as a basis for assessing the fatty acid composition of a milk sample. Naturally, such an approach requires a mapping that takes a spectrum as an input and returns a predicted fatty acid composition as the output (in some fields, such a mapping is called a calibration function). It is clear that the problem of finding such a mapping is a predictive modeling problem, where the input space X is the space of spectra and the output space $Y = \mathbb{S}^q$ (where q is the number of fatty acids).

The collection of the data required to build such a calibration function, as well as the construction of such calibration functions was part of the PhD research of Ivan Stefanov [145]. In the current section, we experiment with a subset of the data that were collected in [145]. In this experiment, we used a dataset containing 75 input-output pairs. These datapoints were obtained by measuring the fatty acid compositions of 75 milk samples (by means of gas chromatography), as well as the Raman spectra (3000 different wave numbers) of these samples. Naturally, these data can be used to build a predictive model that uses the Raman spectrum of a milk sample to predict the fatty acid composition of that sample.

The goal of this section is to illustrate that the inclusion of prior knowledge into this learning problem potentially leads to models with improved predictive performance. As a starting point, we describe the potential sources of prior knowledge that can be used in this setting.

- Output-output knowledge: Due to budgetary constraints, the Raman spectra and accurate fatty acid concentrations could only be determined for about 75 milk samples. However (see also Part II), these samples were selected from a collection of 1033 samples of which a crude estimation of the concentration of

multiple fatty acids is available. Therefore, this extended dataset can be used to obtain a more accurate estimate of the population covariance structure of the outputs.

- Input-output knowledge: The fundamental principles of most spectrophotometric methods are known rather well. As a result, given the molecular structure of a fatty acid of interest, an experienced user of spectrophotometric methods can delineate regions in the spectrum that are informative for determining the concentration of a particular fatty acid. As the molecular structure for the fatty acids of interest is known beforehand, such regions (which are specific for each fatty acid) can be delineated and used within the learning phase by constructing output-specific kernels that focus on these regions.

- Input-input knowledge: The inputs for this learning problem are Raman spectra. Traditionally, spectral data is considered as a type of functional data[5]. This notion typically implies that the inherent dimensionality of the data is much lower than the observed dimensionality (approximately 3000 in this case). This can be taken into account by using kernel functions that are designed for working with spectral data. Examples include wavelet kernels [146] or kernels based on spline smoothing [147].

As argued in the introductory section of this chapter, to transform the predictive modeling of compositional data into a multivariate regression problem, the output space $\mathbb{S}^q$ needs to be transformed to a Euclidean space. Unfortunately, most of the prior knowledge on the output-output relations or input-output relations is available in the original space (which is $\mathbb{S}^q$). Therefore, if this space is transformed, it remains to be seen whether the prior knowledge that is available can be translated to this new space. Interestingly, by choosing a suited set of basis vectors, the ilr-transformation allows such a translation. Indeed, as illustrated in the introductory chapter on compositional data analysis, the ilr-coordinates can often be interpreted as ratios of several components. As these ratios have a clear interpretation, the prior knowledge can at least be partially translated. For example, the input-output knowledge described above can easily be translated. Given two fatty acids with their respective wavelength-regions of interest, the ratio of the concentration of those fatty acids can be assumed to be influenced by the wavelengths in the union of those regions. For this particular example, the output-output knowledge can be translated even more easily. As the population covariance matrix is estimated by

[5] In functional data analysis, it is assumed that each object is characterized by a function. Therefore, the p-dimensional vector that constitutes the observation of an object can be seen as an observation of that function. In case of spectrophotometric data, an object is characterized by its reflectance, emission or transmission spectrum (depending on the technique that is used). The reflectance, emission or transmission that is registered is typically assumed to be a function of the wavelength that is considered. Assuming that there is a continuum of wavelengths, the complete spectrum of an object can be seen as a function that maps a wavelength to an object's reflectance, emission or transmission. The data vector that is observed is a sample of that function.

means of the sample covariance matrix of a (large) sample, the observations in this sample can simply be transformed into the new space, prior to the computation of the covariance matrix.

Unfortunately, the prior knowledge that was available at the writing of this dissertation does not allow the inclusion of all types of prior knowledge. Therefore, we limit this discussion to a setting where only output-output knowledge is incorporated. Moreover, the concentration of a considerable number of fatty acids seems to be extremely hard to predict using the dataset that was available (this was also concluded in [145]). Additionally, a large number of the fatty acids that were available in the training dataset were not reported in the large dataset (of 1033 observations). To obtain a setting that can be used to illustrate our approach, out of the original datasets, only 10 fatty acids were retained (the selection is based on [66] p. 93, the selected fatty acids are listed in Appendix 11.B). Subsequently, the ilr-transformation was used to transform the compositional vectors into a Euclidean space. The matrix of basis vectors that was used is presented in Appendix 11.B. Unfortunately, from several preliminary experiments in which independent models were fit to the individual transformed outputs, it turned out that only 3 of the 9 outputs (output nrs 6, 7 and 9) could be modeled. For the remaining outputs, there was close to no link between the spectrum that was measured and the output (this was decided based on visual inspection of plots of the predicted output versus the observed output). Therefore, the learning experiment was further narrowed down to those three outputs.

To investigate the potential advantage of incorporating prior knowledge on output-output correlations in this problem setting, the framework of kernel-based vector-functions was used. To fit the models, the empirical L_2-regularized squared loss was minimized. Moreover, we used a separable kernel of the form $\mathbf{K}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}' \mathbf{B}$, where $\mathbf{B}$ was choosen according to one of the following strategies:

1. $\mathbf{B} = \mathbf{I}_3$,
2. $\mathbf{B}$ is the sample covariance matrix of the large "prior knowledge" dataset containing 897 points. Only 897 from a total of 1033 datapoints were retained. A number of datapoints were deleted as for those datapoints, the reported concentration of at least one fatty acid was zero. Moreover, as the training dataset did not contain zero-entries, the zeros in the "prior knowledge" dataset are probably rounding zeros or measurement errors.
3. $\mathbf{B}$ is the sample covariance matrix of the large training dataset.

To obtain a training dataset, 60 input-output pairs were selected randomly (out of the total of 75 points) (the training set). The remaining 15 points were used to assess the performance of the model (the test set). The performance measure was the sum of squared Euclidean distances (in the ilr-transformed space) between the predictions and the observed values in the test set. To tune the complexity

Table 11.2: Median, mean and standard deviation of the performances (expressed as the sum of squared errors) obtained by re-sampling 100 train and test sets.

	Setting 1	Setting 2	Setting 3
Median (of squared errors)	12.9140	11.4521	11.7179
Mean (of squared errors)	16.6222	15.5561	15.6770
Standard deviation	11.3570	10.8612	10.9209

Table 11.3: Pairwise comparison of the three settings. The reported number represents the number of times that a given setting outperforms another setting over 100 runs, when the performance is expressed by means of the sum of squared errors.

	Number of times (out of 100)
Setting 2 outperforms setting 1	82
Setting 3 outperforms setting 1	90
Setting 2 outperforms setting 3	68

parameter λ, a 5-fold cross validation strategy was used. This process was repeated 100 times. Table 11.2 shows the median, the mean and the standard deviation of the test error over 100 runs. Additionally, Table 11.3 reports pairwise comparisons between the three strategies. From these tables, it can be seen that strategies 2 and 3 outperform the case where models are learned independently ($\mathbf{B} = \mathbf{I}_3$). This can be seen in both tables. The median performance of both methodologies is better than the case where $\mathbf{B} = \mathbf{I}_3$ However, the size of the improvement is rather limited. Moreover, from 100 runs both strategies outperform the first setting in over 80% of the cases. On the other hand, based on these experiments, it can be expected that the setting with prior knowledge (setting 2) mostly will outperform the strategy where $\mathbf{B}$ equals the sample covariance matrix of the training dataset. Given the limited size of the dataset, it is hard to draw statistically relevant conclusions here. Nevertheless, we cautiously conclude that the incorporation of prior knowledge can improve the predictive capabilities of the resulting model.

11.7. Conclusions and discussion

11.7.1. Predicting compositional data and prior knowledge

In this chapter, the ilr-transform was used to transform the problem of predicting compositional data into a (more traditional) multivariate regression problem. This transformation allows a plenitude of multivariate regression methods to be used that are dedicated to exploiting dependencies between multiple outputs to improve the predictive performance of a model. More precisely, we focused on the incorporation of prior knowledge when learning kernel-based vector functions. The

main contributions of this chapter are (1) the specification of three types of prior knowledge, (2) the development of a kernel that can incorporate these types of knowledge into a learning problem, (3) a (partial) theoretical analysis of the effect of incorporating prior knowledge and (4) the development of a computationally tractable optimization procedure that can be used during the learning phase.

From a theoretical analysis, it follows that for separable kernels of the form $\mathbf{K}(\mathbf{x}, \mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x})\,\mathbf{B}$, choosing $\mathbf{B} = \mathrm{cov}(\mathcal{Y})$ is a good choice when minimizing the expected risk. This observation suggests that the incorporation of information on the (population) covariance structure in a separable kernel can lead to an improvement of the predictive performance of the models that are learned. This observation reappears in the experiments, suggesting that the methodology that is proposed can lead to improved predictive models. It should be noted, however, that the increase in predictive power is rather limited. Naturally, the importance of such an improvement depends on the application at hand. Additionally, one should be careful when drawing conclusions about the potential of an approach based on only one dataset. With respect to that, more empirical experiments are needed (on different datasets) to confirm these results.

Experiments on artificially generated data illustrate that the inclusion of input-output relations can lead to models with improved predictive performance. Mainly in data-scarce settings, this type of prior knowledge can be advantageous. Moreover, this experiment suggests that the kernel that was proposed can effectively incorporate the three types of prior knowledge that were described. However, it remains unclear how these types of prior knowledge interact. For example, given two outputs that seem to be correlated (assume we have prior knowledge on this relation), on the other hand, prior knowledge on input-output relations suggests that these outputs depend on two disjoint sets of inputs. This situation can naturally occur when the inputs are correlated. It remains to be seen here how both types of prior knowledge can be optimally combined, as at first sight they may seem to steer the learning process in opposite directions. It is likely that the issues that are discussed here will appear in several situations. Moreover, the study of these issues may lead to more insights in the general learning problem.

11.7.2. Alternative ways to incorporate prior knowledge

We continue this discussion with some alternatives to the incorporation of prior knowledge in multivariate regression problems. As stated before, choosing $\mathbf{K}(\mathbf{x}, \mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x})\,\mathbf{B}$, with $\mathbf{B} = \mathrm{cov}(\mathcal{Y})$ seems to be beneficial for the predictive performance of the resulting model. Indeed, we have some theoretical guarantees that choosing $\mathbf{B} = \mathrm{cov}(\mathcal{Y})$ is not bad. On the other hand, there may exist other manners to define $\mathbf{B}$ that outperform the current approach. Perhaps, a more profound study of the expected risk could lead to superior approaches.

To conclude this discussion, we briefly elaborate on a very intuitive way to incorporate prior knowledge on input-input correlations and output-output correlations. To that end, consider the idealized setting where output $\mathcal{Y}$ is a q-dimensional random vector and the input $\mathcal{X}$ is a p-dimensional random vector. Moreover, assume that we know (this is our prior knowledge) that $\mathrm{cov}(\mathcal{Y}) = \Sigma_{\mathrm{Y}}$ and $\mathrm{cov}(\mathcal{X}) = \Sigma_{\mathrm{X}}$. The goal exists of learning a linear function $\mathbf{f}$, with parameter matrix $\mathbf{A}$ such that $\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x}$.

To ensure that $\mathbf{A}$ satisfies the prior knowledge, the following constraint can be enforced:

$$\mathrm{cov}(\mathbf{A}\mathcal{X}) = \Sigma_{\mathrm{Y}}\,.$$

This constraint can be rewritten as $\mathbf{A}\Sigma_{\mathrm{X}}\mathbf{A}^{\top} = \Sigma_{\mathrm{Y}}$.

This leads to the following optimization problem:

$$\begin{aligned} \underset{\mathbf{A}\in\mathbb{R}^{p\times p}}{\text{minimize}} \quad & \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{A}\mathbf{x}_i - \mathbf{y}_i\|_2^2 + \lambda\sum_{j=1}^{q}\|\mathbf{A}_{j,\cdot}\|_2^2 \\ \text{subject to} \quad & \mathbf{A}\Sigma_{\mathrm{X}}\mathbf{A}^{\top} = \Sigma_{\mathrm{Y}} \end{aligned}$$

Unfortunately, the equality constraint in this optimization problem is highly nonlinear (see Appendix 11.C for a discussion on how this optimization problem can be solved). Nevertheless, the problem formulation itself provides a very intuitive way of incorporating prior knowledge. As opposed to the method we proposed to incorporate prior knowledge when learning vector-valued functions, this approach forms a direct way of enforcing the prior knowledge that is given. More precisely, the resulting model will respect the prior knowledge whereas the methodology proposed earlier will only guide the learning phase.

11.A. Computation of the expected risk

In this appendix, we derive the formula in Eq. (11.11) to compute $\mathbb{E}[r(\hat{\mathbf{f}}_i)]$. We start from the following statistical model:

$$\mathcal{Y} = \mathbf{A}\mathcal{X} + \mathcal{E}\,, \tag{11.15}$$

where $\mathrm{cov}(\mathcal{X}) = \Sigma$, $\mathbb{E}[\mathcal{X}] = \mathbf{0}_q$ $\mathrm{cov}(\mathcal{E}) = \sigma^2\mathbf{I}_q$ and $\mathbb{E}[\mathcal{E}] = \mathbf{0}_q$.

However, as we will perform a fixed design analysis, the inputs are considered fixed. More precisely, we assume to have been given n input vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ such that $\frac{1}{n}\mathbf{X}^{\top}\mathbf{X} = \Sigma$. In this case, the model reduces to

$$\mathcal{Y}^i = \mathbf{A}\mathbf{x}_i + \mathcal{E}^i\,, \tag{11.16}$$

where $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_q)^\top$, $\mathrm{cov}(\mathcal{E}^i) = \sigma^2 \mathbf{I}_q$ and all $\mathcal{E}^i$ are independent. It should be noted that as usual, a single *observed* training data set T is denoted $T = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n = (\mathbf{X}, \mathbf{Y})$. Moreover, we denote the matrix of error terms as $\mathbf{E} = \mathbf{Y} - \mathbf{X}\mathbf{A}^\top$. Notably, when re-sampling a dataset, the inputs $\mathbf{X}$ persist but the outputs and the error terms are renewed.

11.A.1. Preliminaries

Univariate case

As a starting point, let us consider the single-output case (taking $q = 1$) and assume that the underlying statistical model is

$$\mathcal{Y}^i = \mathbf{a}^\top \mathbf{x}_i + \mathcal{E}^i \,, \tag{11.17}$$

where $\mathbf{a} \in \mathbb{R}^p$.

Now, let $\hat{\mathbf{a}}$ be the minimizer of the L_2-regularized squared loss (Eq. (11.1)). This means that $\hat{\mathbf{a}} = (\mathbf{X}^\top \mathbf{X} + n\,\lambda \mathbf{I}_p)^{-1} \mathbf{X}^\top \mathcal{Y}$ (this is called the ridge estimate)[6]. Moreover, let $\hat{f}(\mathbf{x}) = \hat{\mathbf{a}}^\top \mathbf{x}$, it is well known that (see for instance [140])

$$\mathbb{E}[\hat{\mathbf{a}}] = \mathrm{Diag}\left(\frac{\gamma_k}{\gamma_k + \lambda}\right) \mathbf{a} \,, \tag{11.18}$$

and

$$\mathbb{E}[r(\hat{f})] = \overbrace{\mathbb{E}\left[\|\hat{\mathbf{a}} - \mathbb{E}[\hat{\mathbf{a}}]\|_\Sigma^2\right]}^{\text{variance}} + \overbrace{\|\mathbb{E}[\hat{\mathbf{a}}] - \mathbf{a}\|_\Sigma^2}^{\text{bias}} \tag{11.19}$$

$$= \frac{\sigma^2}{n} \sum_{i=j}^{p} \left(\frac{\gamma_j}{\gamma_j + \lambda}\right)^2 + \sum_{j=1}^{p} a_i^2 \frac{\gamma_j}{(1 + \gamma_j/\lambda)^2} \,. \tag{11.20}$$

Equivalently, we can minimize the L_2-regularized squared loss in the dual form, given a kernel k and an implied hypothesis space H:

$$\tilde{f} = \arg\min_{f \in H} \frac{1}{n} \sum_{i=1}^{n} (f(\mathbf{x}_i) - \mathcal{Y}^i)^2 + \lambda \, \|f\|_H^2 \,.$$

When $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$ (this is the linear kernel), it can be shown (see for instance [148]) that $\tilde{f} = \hat{f}$. Therefore, we can restrict the analysis to the primal form.

The multivariate case

[6] It must be stressed that $\hat{\mathbf{a}}$ is a random vector (even though we do not use the random vector notation here).

We now turn our attention to the multivariate case. Interestingly, Baldassarre *et al.* [141] have shown that for a matrix-valued kernel of the form $\mathbf{K}(\mathbf{x}, \mathbf{x}') = \kappa(\mathbf{x}, \mathbf{x}')\mathbf{B}$ (where $\mathbf{B}$ is a positive definite matrix) and a dataset $(\mathbf{X}, \mathbf{Y})$, the minimizer $\hat{\mathbf{f}}$ of

$$\frac{1}{n}\sum_{i=1}^{n} \|\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i\|_2^2 + \lambda \|\mathbf{f}\|_H^2 \ ,$$

can be computed efficiently using the following procedure (where we immediately use $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}$):

1. Compute the eigendecomposition of $\mathbf{B}$, *i.e.* $\mathbf{B} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$, such that $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_q)$ and $\mathbf{u}_i$ is the ith eigenvector of $\mathbf{B}$ and $\mathbf{D}$ is a diagonal matrix such that $\mathbf{D}_{i,i}$ is the ith eigenvalue of $\mathbf{B}$
2. Compute the transformed output matrix $\tilde{\mathbf{Y}} = \mathbf{Y}\mathbf{U}^\top$.
3. Compute $\tilde{\mathbf{A}} = (\tilde{\mathbf{a}}_1, \ldots, \tilde{\mathbf{a}}_q)^\top$ where

$$\tilde{\mathbf{a}}_j = (\mathbf{X}^\top \mathbf{X} + n\lambda/\mathbf{D}_{j,j}\, \mathbf{I}_p)^{-1} \mathbf{X}^\top \underbrace{\mathbf{Y}\mathbf{u}_k}_{\tilde{\mathbf{Y}}_{\cdot,j}} \ . \tag{11.21}$$

4. Compute $\hat{\mathbf{A}} = \mathbf{U}\tilde{\mathbf{A}}$. Let $\hat{\mathbf{f}}(\mathbf{x}) = \hat{\mathbf{A}}\mathbf{x}$.

Interestingly, this computational trick shows that the application of the multivariate ridge regression estimate implicitly leads to q uni-variate ridge estimates in a transformed space. This analogy will be used to derive a formula to compute the expected risk.

11.A.2. Multivariate expected risk

As in the univariate case, we will be working in a fixed design setting. Therefore, we assume the following statistical model:

$$\mathcal{Y}^i = \mathbf{A}\mathbf{x}_i + \mathcal{E}^i \ , \tag{11.22}$$

where $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_q)^\top$, $\frac{1}{n}\mathbf{X}^\top\mathbf{X} = \Sigma$ and $\mathrm{cov}(\mathcal{E}^i) = \sigma^2 \mathbf{I}_q$ (and all $\mathcal{E}^i$ are independent and normally distributed).

Computation of the variance

We first focus on computing the variance associated with the ith output. More precisely, we compute

$$\mathbb{E}\left[\|\hat{\mathbf{a}}_i - \mathbb{E}[\hat{\mathbf{a}}_i]\|_\Sigma^2\right] \ .$$

As a starting point of our analysis, let us reconsider steps 3 and 4 of the procedure presented above. From steps 3 and 4, we have that $\hat{\mathbf{a}}_i = \sum_{j=1}^{q} \mathbf{U}_{i,j}\tilde{\mathbf{a}}_j$.

$$\begin{aligned}\mathbb{E}\left[\|\hat{\mathbf{a}}_i - \mathbb{E}[\hat{\mathbf{a}}_i]\|_\Sigma^2\right] &= \mathbb{E}\left[\left\|\sum_{j=1}^{q}(\tilde{\mathbf{a}}_j - \mathbb{E}[\tilde{\mathbf{a}}_j])\mathbf{U}_{i,j}\right\|_\Sigma^2\right] \\ &= \sum_{j=1}^{q}\mathbb{E}[\|\tilde{\mathbf{a}}_j - \mathbb{E}[\tilde{\mathbf{a}}_j]\|]_\Sigma^2 \\ &\qquad +2\sum_{k<l}\mathbf{U}_{i,k}\mathbf{U}_{i,l}\mathbb{E}[(\tilde{\mathbf{a}}_k - \mathbb{E}[\tilde{\mathbf{a}}_k])^\top\Sigma(\tilde{\mathbf{a}}_l - \mathbb{E}[\tilde{\mathbf{a}}_l])^\top]\,.\end{aligned}$$

We now show that $\mathbb{E}[(\hat{\mathbf{a}}_k - \mathbb{E}[\hat{\mathbf{a}}_k])^\top\Sigma(\hat{\mathbf{a}}_l - \mathbb{E}[\hat{\mathbf{a}}_l])] = 0$ for $k \neq l$.

Combining Eq. (11.21) and $\mathbf{Y} = \mathbf{X}\mathbf{A}^\top + \mathbf{E}$, it can easily be seen that $\mathbb{E}[\hat{\mathbf{a}}_k] = (\mathbf{X}^\top\mathbf{X} + n\lambda/\mathbf{D}_{j,j}\,\mathbf{I}_p)^{-1}\mathbf{X}^\top\mathbf{X}\mathbf{A}^\top\mathbf{u}_k$. Moreover, we have that

$$\tilde{\mathbf{a}}_k = \mathbb{E}[\tilde{\mathbf{a}}_k] + (\mathbf{X}^\top\mathbf{X} + n\,\lambda/\mathbf{D}_{j,j}\,\mathbf{I}_p)^{-1}\mathbf{X}^\top\mathbf{E}\mathbf{u}_k\,.$$

It was assumed that $\mathbf{E}$ is a matrix of independent normally distributed random variables. Using this assumption, and due to the orthogonality of $\mathbf{u}_k$ and $\mathbf{u}_l$, the random vectors $\mathbf{E}\mathbf{u}_k$ and $\mathbf{E}\mathbf{u}_\ell$ are independent. Moreover $\mathbb{E}[\mathbf{E}\mathbf{u}_k] = \mathbb{E}[\mathbf{E}\mathbf{u}_l] = \mathbf{0}_n$. Now let

$$\mathbf{e}_k = (\mathbf{X}^\top\mathbf{X} + n\,\lambda/\mathbf{D}_{j,j}\,\mathbf{I}_p)^{-1}\mathbf{X}^\top\mathbf{E}\mathbf{u}_k\,.$$

It is easy to see that $\mathbf{e}_k$ and $\mathbf{e}_l$ (where $k \neq l$) are independent random vectors and $\mathbb{E}[\mathbf{e}_k] = \mathbb{E}[\mathbf{e}_l] = \mathbf{0}_p$. The identities above are combined to obtain the following result:

$$\mathbb{E}[(\hat{\mathbf{a}}_k - \mathbb{E}[\hat{\mathbf{a}}_k])^\top\Sigma\,(\hat{\mathbf{a}}_l - \mathbb{E}[\hat{\mathbf{a}}_l])] = \mathbb{E}[\mathbf{e}_k^\top\Sigma\,\mathbf{e}_l] = 0\,.$$

Combining this result with Eq. (11.20), we obtain that

$$\mathbb{E}\left[\|\hat{\mathbf{a}}_i - \mathbb{E}[\hat{\mathbf{a}}_i]\|_\Sigma^2\right] = \sum_{j=1}^{q}\mathbb{E}[\|\tilde{\mathbf{a}}_j - \mathbb{E}[\tilde{\mathbf{a}}_j]\|]_\Sigma^2 \tag{11.23}$$

$$= \frac{\sigma^2}{n}\sum_{j=1}^{q}\mathbf{U}_{i,j}^2\sum_{k=1}^{p}\left(\frac{\gamma_k}{\gamma_k + \frac{\lambda}{\mathbf{D}_{j,j}}}\right)^2 \tag{11.24}$$

Computation of the bias

Now let $\mathbf{A}^\bullet = \mathbf{U}^\top\mathbf{A}$ and $\mathbf{A}^\bullet = (\mathbf{a}_1^\bullet, \ldots, \mathbf{a}_q^\bullet)^\top$. It can easily be seen that $\mathbf{a}_i =$

$\sum_{j=1}^{q} \mathbf{U}_{i,j}\mathbf{a}_j^{\bullet}$. Therefore, we have that

$$\|\mathbb{E}[\hat{\mathbf{a}}_i] - \mathbf{a}_i\|_{\Sigma}^2 = \left\| \sum_{j=1}^{q} (\mathbb{E}[\tilde{\mathbf{a}}_j] - \mathbf{a}_j^{\bullet})\mathbf{U}_{i,j} \right\|_{\Sigma}^2 . \tag{11.25}$$

As $\tilde{\mathbf{a}}_j$ can be seen as the ridge estimate of $\mathbf{a}_j^{\bullet}$, it follows from Eq. (11.18) that

$$\mathbb{E}[\tilde{\mathbf{a}}_j] = \mathrm{Diag}\left(\frac{\gamma_k}{\gamma_k + \frac{\lambda}{\mathbf{D}_{j,j}}} \right) \mathbf{a}_j^{\bullet} .$$

Using $\mathbf{a}_j^{\bullet} = \mathbf{A}^{\top}\mathbf{u}_j$, Eq. (11.25) can be rewritten as follows:

$$\|\mathbb{E}[\hat{\mathbf{a}}_i] - \mathbf{a}_i\|_{\Sigma}^2 = \left\| \sum_{j=1}^{q} \mathrm{Diag}\left(\frac{\gamma_k}{\gamma_k + \frac{\lambda}{\mathbf{D}_{j,j}}} - 1 \right) \mathbf{A}^{\top}\mathbf{u}_j\mathbf{U}_{i,j} \right\|_{\Sigma}^2 . \tag{11.26}$$

Moreover, as $\Sigma = \mathrm{Diag}(\gamma_k)$, we have that:

$$\|\mathbb{E}[\hat{\mathbf{a}}_i] - \mathbf{a}_i\|_{\Sigma}^2 = \sum_{k=1}^{p} \gamma_k \left(\sum_{j=1}^{q} \mathrm{Diag}\left(\frac{\gamma_k}{\gamma_k + \frac{\lambda}{\mathbf{D}_{j,j}}} - 1 \right) \mathbf{A}^{\top}\mathbf{u}_j\mathbf{U}_{i,j} \right)^2 . \tag{11.27}$$

Combining Eqs. (11.24) and (11.27) leads to Eq. (11.11).

11.B. Computation of an orthogonal basis

The following fatty acids were retained: iso C14:0, iso C15:0, anteiso c15:0, iso c16:0, iso c17:0, c17:0, trans10 c18:1, trans 11 c18:1, c 9 t 11 c18:2, c15:0.

The following matrix $\mathbf{\Phi}$ was used to obtain the ilr-transformed data (using the reasoning in Section 2.4, this matrix can be used to obtain an orthogonal basis E).

	1	2	3	4	5	6	7	8	9
iso C14:0	-0.707	-0.408	-0.289	-0.224	-0.183	-0.154	-0.134	-0.118	-0.105
iso C15:0	0.707	-0.408	-0.289	-0.224	-0.183	-0.154	-0.134	-0.118	-0.105
anteiso c15:0	0.000	0.816	-0.289	-0.224	-0.183	-0.154	-0.134	-0.118	-0.105
iso c16:0	0.000	0.000	0.866	-0.224	-0.183	-0.154	-0.134	-0.118	-0.105
iso c17:0	0.000	0.000	0.000	0.894	-0.183	-0.154	-0.134	-0.118	-0.105
c17:0	0.000	0.000	0.000	0.000	0.913	-0.154	-0.134	-0.118	-0.105
trans10 c18:1	0.000	0.000	0.000	0.000	0.000	0.926	-0.134	-0.118	-0.105
trans 11 c18:1	0.000	0.000	0.000	0.000	0.000	0.000	0.935	-0.118	-0.105
c 9 t 11 c18:2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.943	-0.105
c15:0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.949

11.C. Encoding prior knowledge via geometrically constrained programming

Clearly, the following optimization problem is not convex.

$$\begin{aligned} \underset{\mathbf{A}\in\mathbb{R}^{p\times p}}{\text{minimize}} \quad & \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{A}\mathbf{x}_i-\mathbf{y}_i\|_2^2+\lambda\sum_{j=1}^{q}\|\mathbf{A}_{j,.}\|_2^2 \\ \text{subject to} \quad & \mathbf{A}\Sigma_{\mathrm{X}}\mathbf{A}^\top=\Sigma_{\mathrm{Y}} \end{aligned}$$

As a result, globally solving it is probably hard. Nevertheless, it can be attempted to find a local optimum to this problem. Hereafter, we will assume that $p=q$ (however, this can probably be generalized). Unfortunately, the structure of this problem prevents traditional local solvers (such as for instance sequential quadratic programming solvers) from being used. This is mainly due to the fact that the equality constraint can not be linearized (or at least the linearized form deviates strongly from the original form). However, we can rewrite the equality constraint using the eigendecomposition of the matrices Σ_{X} and Σ_{Y}. Let $\mathbf{P}_{\mathrm{X}}$ (resp. $\mathbf{P}_{\mathrm{Y}}$) be the matrix of eigenvectors of Σ_{X} (resp. Σ_{X}), and $\mathbf{D}_{\mathrm{X}}$ (resp. $\mathbf{D}_{\mathrm{X}}$) be a diagonal matrix containing the eigenvalues of Σ_{X} (resp. Σ_{Y}), *i.e.* $\Sigma_{\mathrm{X}}=\mathbf{P}_{\mathrm{X}}\mathbf{D}_{\mathrm{X}}\mathbf{P}_{\mathrm{X}}^\top$ (resp. $\Sigma_{\mathrm{Y}}=\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}\mathbf{P}_{\mathrm{Y}}^\top$). We now have that

$$\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)\Sigma_{\mathrm{X}}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)^\top=\mathbf{I}_p\,, \tag{11.28}$$

and

$$\Sigma_{\mathrm{Y}}=\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)\mathbf{I}_p\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)^\top. \tag{11.29}$$

Now, let $\mathbf{U}$ be a $p\times p$ orthonormal matrix. From Eq. (11.28), we have that

$$\mathbf{U}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)\Sigma_{\mathrm{X}}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)^\top\mathbf{U}^\top=\mathbf{I}_p\,.$$

Combining this with Eq. (11.29) and using $\mathbf{P}_{\mathrm{X}}^{-1} = \mathbf{P}_{\mathrm{X}}^{\top}$ and $\mathbf{P}_{\mathrm{Y}}^{-1} = \mathbf{P}_{\mathrm{Y}}^{\top}$, we obtain that

$$\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)\mathbf{U}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)\Sigma_{\mathrm{X}}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)^{\top}\mathbf{U}^{\top}\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)^{\top} = \Sigma_{\mathrm{Y}}\,,$$

or, equivalently,

$$\underbrace{\left(\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)\mathbf{U}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)\right)}_{\mathbf{A}}\Sigma_{\mathrm{X}}\underbrace{\left(\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)\mathbf{U}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)\right)^{\top}}_{\mathbf{A}^{\top}} = \Sigma_{\mathrm{Y}}$$

In summary, we can let the matrix $\mathbf{U}$ be the optimization variable with the constraint that $\mathbf{U}$ is an orthonormal matrix, and define $\mathbf{A} = \left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)\mathbf{U}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)$. This leads to the following optimization problem

$$\begin{aligned}
\underset{\mathbf{A}\in\mathbb{R}^{p\times p}}{\text{minimize}} \quad & \frac{1}{n}\sum_{i=1}^{n}\left\|\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)\mathbf{U}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)\mathbf{x}_i - \mathbf{y}_i\right\|_2^2 \\
& \qquad + \lambda\sum_{j=1}^{q}\left\|\left(\left(\mathbf{P}_{\mathrm{Y}}\mathbf{D}_{\mathrm{Y}}^{\frac{1}{2}}\right)\mathbf{U}\left(\mathbf{D}_{\mathrm{X}}^{-\frac{1}{2}}\mathbf{P}_{\mathrm{X}}^{-1}\right)\right)_{j,\cdot}\right\|_2^2 \\
\text{subject to} \quad & \mathbf{U}\,\mathbf{U}^{\top} = \mathbf{I}_p
\end{aligned}$$

The equality constraint $\mathbf{U}\,\mathbf{U}^{\top} = \mathbf{I}_p$ is known as a geometric constraint [149], expressing that the optimization variable should belong to the manifold of orthonormal matrices. Even though geometrically constrained programming is a rather recent area of research, several algorithms have been developed that can be proven to converge to a local optimum [149, 150].

12 Predictive modeling of compositional outputs with ordered components

12.1. Introduction

We start this chapter with a motivating (real life) problem setting in agriculture. Head blight is a fungal disease in cereal crops that has a considerable economic impact [151]. Therefore, this disease has been studied rather extensively during the past decades. It is known that the infection degree of a field with head blight is strongly influenced by the micro-climate during the flowering season. As a result, information on the micro-climate of a wheat field could be used to build a model that can be used to assess the expected degree of infection of that field. The setting described here naturally leads to a predictive modeling problem where the goal is to predict the infection degree of a field (the output) using the information of the micro-climate on that field (the input). Naturally, such a predictive modeling problem requires a thorough description of the degree of infection of a field. Such a description is presented in [152]. The authors presented a dataset that describes the degree of infection of cereal crops with head blight disease in Flanders (Belgium). To describe the degree of infection of a wheat field, a number of ears are selected randomly from the field. Each ear is classified by an expert in one of five ordinal classes: not infected, slightly infected, moderately infected, heavily infected, fully infected. As such, the fractions of ears in each class constitute a compositional vector describing the infection degree of the field. As inputs to this predictive modeling problem, 45 micro-climatological variables were recorded at each field.

In this chapter, we develop a class of predictive models that can be used to solve prediction problems such as the one above. In a first attempt, we could neglect the ordinal nature of the classes, use the isometric log-ratio transform and solve the problem as a multivariate regression problem using the techniques described in the previous chapter. However, in the current chapter we investigate several alternative approaches that can be used to solve this problem. The remainder of this chapter is organized as follows:

- In Section 12.2, we elaborate on the use of multinomial deviance as a loss function.
- In Section 12.3, models for compositional data with ordered components are introduced.
- In Section 12.4, the added value of the proposed methodology is illustrated by means of an extensive set of experiments on artificially generated data.

- In Section 12.5, the proposed methodology is illustrated on a real-life problem setting.
- In Sections 12.6 and 12.7, a relaxation of the initial model is proposed.
- In Section 12.8, concluding remarks are formulated.

12.2. Multinomial deviance for predicting compositional data

As a starting point, we focus on constructing predictive models that do not take the order relation on the components into account. Therefore, assume that we have been given a training dataset $T = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ with $\mathbf{y}_i \in \mathbb{S}^q$. Recall from Chapter 11 that the hypothesis space H can be a set of functions $\mathbf{f} : \mathbb{R}^p \to \mathbb{S}^q$ that can be written in the following form:

$$\begin{aligned} f_k(\mathbf{x}) &= \frac{\exp(\mathbf{w}_k^\top \mathbf{x} + b_k)}{1 + \sum_{k=1}^{q-1} \exp(\mathbf{w}_k^\top \mathbf{x} + b_k)}\,, \qquad \text{for } k = 1, \ldots, q-1\,, \\ f_q(\mathbf{x}) &= \frac{1}{1 + \sum_{k=1}^{q-1} \exp(\mathbf{w}_k^\top \mathbf{x} + b_k)}\,, \end{aligned} \tag{12.1}$$

where $\mathbf{w}_k \in \mathbb{R}^p$ and $b_k \in \mathbb{R}$. Moreover, this formulation allows for the following complexity criterion $c(\mathbf{f}) = \sum_{k=1}^{q-1} \|\mathbf{w}_k\|_2^2$. When using the multinomial deviance as a loss function, *i.e.* $l(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{k=1}^{q} y_k \log(\hat{y}_k)$, the following regularized empirical risk function is obtained

$$r(\mathbf{f}) = \sum_{i=1}^{n} \sum_{k=1}^{q} y_{i,k} \log(f_k(\mathbf{x}_i)) + \lambda \sum_{k=1}^{q-1} \|\mathbf{w}_k\|_2^2\,. \tag{12.2}$$

Clearly, this regularized empirical risk function is (almost) identical to the risk function for multi-class logistic regression [122, 153, 154] which is very popular in multi-class classification. The only difference here is that $\mathbf{y}_i \in \mathbb{S}^q$, whereas $\mathbf{y}_i \in \{0,1\}^q$ (more precisely, a vector containing $q-1$ zeros and a one at the position that represents the class to which the instance belongs) in the classification setting. There is a simple link between both settings. To show this, we create a new dataset $T^* = \{\mathbf{x}_{i,j}^*, \mathbf{y}_{i,j}^*\}_{i=1,j=1}^{n,\tau}$ that contains τ duplicates of each instance in T. In this new dataset, we have that $\mathbf{x}_{i,j}^* = \mathbf{x}_i$ for all i and j. The outputs are elements of $\{0,1\}^q$ that are assigned such that they respect the proportions given in $\mathbf{y}_i$. More precisely, we require that

$$y_{i,k} \approx \frac{1}{\tau} \sum_{j=1}^{\tau} y_{i,j,k}^*\,.$$

This relation shows that, from a methodological point of view, both methods are strongly related. Moreover, this similarity suggests that numerical procedures used to fit multi-class logistic regression models can be used to find the minimizer of $r(\mathbf{f})$. However, it should be noted here that the size of T^* can be rather large, leading to intractable optimization problems. However, some approaches such as the sequential minimal optimization approach of [153, 155] can trivially be extended to handle compositional vectors directly.

Using the same analogy, the resulting method can easily be *kernelized*. Therefore, let g_k, $k = 1, \ldots, q-1$, be a linear function such that $g_k(\mathbf{x}) = \mathbf{w}_k^\top \mathbf{x}$. These functions can be used to rewrite Eq. (12.1):

$$\begin{aligned} f_k(\mathbf{x}) &= \frac{\exp(g_k(\mathbf{x}) + b_k)}{1 + \sum_{k=1}^{q-1} \exp(g_k(\mathbf{x}) + b_k)}\,, \qquad \text{for } k = 1, \ldots, q-1\,, \\ f_q(\mathbf{x}) &= \frac{1}{1 + \sum_{k=1}^{q-1} \exp(g_k(\mathbf{x}) + b_k)}\,, \end{aligned}$$

More generally, given a kernel $\kappa : X \times X \to \mathbb{R}$, a reproducing kernel Hilbert space H of functions can be constructed (following the methodology of Chapter 11). Following [133, 154], the minimizer of the regularized empirical loss function

$$r(\mathbf{f}) = \sum_{i=1}^{n} \sum_{k=1}^{q} y_{i,k} \log(f_k(\mathbf{x}_i)) + \lambda \sum_{k=1}^{q-1} \|g_k\|_H^2\,,$$

admits a representation of the form $\hat{g}_k(\mathbf{x}) = \sum_{i=1}^{n} \alpha_{i,k} \kappa(\mathbf{x}, \mathbf{x}_i)$.

12.3. Predicting compositional data with ordered components

The previous section illustrates the close link between multi-class classification models and predictive models for compositional data that use multinomial deviance as a loss function. As ordinal regression can be seen as a special case of multi-class classification, it is worth investigating whether ideas from the field of ordinal regression can be borrowed to construct predictive models for compositional data with ordered components. Therefore, we elaborate on ordinal regression models hereafter.

12.3.1. Models for ordinal regression problems

In a multi-class classification setting, the goal is to learn a mapping from an input space X to a finite set $C = \{C_1, \ldots, C_q\}$ containing q labels. To this end, each

object is usually represented by a p-dimensional feature vector $\mathbf{x} \in X$ and a class label $y \in C$. A training dataset T of n i.i.d. observations can then be denoted as a set of couples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,p})^\top$, in which we assume that the couples $(\mathbf{x}_i, y_i)$ are realizations of the random vector $(\mathcal{X}, \mathcal{Y})$. Moreover, using this notation, an ordinal regression problem can be seen as a special case of a classification problem where the label set is endowed with a linear order relation $C_1 \prec C_2 \prec \cdots \prec C_q$.

Ordinal regression has been studied quite extensively in statistics [122, 156]. This problem can be seen as a specific case of the general multi-class classification problem and thus be modeled with the same techniques. However, multi-class classification methods neglect the order endowed on the label set. Several types of ordinal regression models are capable of taking this order into account, such as adjacent-categories models, continuation-ratio logit models and latent variable models [122]. Within each of these types, the order on the labels is looked upon from a different perspective. Latent variable models are probably the most important type of ordinal regression models [157, 158, 159, 160, 161]. These models motivate the ordinal scale as the result of coarse measurements of a continuous variable, called the latent variable. It is typically assumed that the latent variable is difficult to measure or cannot be observed itself. This type of models can be represented in the following general form

$$f(\mathbf{x}) = \begin{cases} C_1 & \text{, if } g(\mathbf{x}) \leq \theta_1 \,, \\ C_2 & \text{, if } \theta_1 < g(\mathbf{x}) \leq \theta_2 \,, \\ \vdots & \\ C_q & \text{, if } \theta_{q-1} < g(\mathbf{x}) \,, \end{cases} \tag{12.3}$$

with $g : X \to \mathbb{R}$ the function that models the latent variable and $\theta_1 < \ldots < \theta_{q-1}$ a set of thresholds. Therefore, fitting ordinal regression models boils down to estimating a function g and a set of thresholds.

The proportional odds model [162] is in statistics without doubt the best known and most applied model to represent ordinal responses. It is naturally derived from the latent variable motivation. As a starting point, this model assumes that the relationship between the random variable $\mathcal{Y}$ that models the output and the random input vector $\mathcal{X}$ is the following:

$$\mathcal{Y} = \begin{cases} C_1 & \text{, if } g(\mathcal{X}) + \mathcal{E} \leq \theta_1 \,, \\ C_2 & \text{, if } \theta_1 < g(\mathcal{X}) + \mathcal{E} \leq \theta_2 \,, \\ \vdots & \\ C_q & \text{, if } \theta_{q-1} < g(\mathcal{X}) + \mathcal{E} \,, \end{cases} \tag{12.4}$$

where $\mathcal{E}$ is an error term that follows a logistic distribution with mean 0. Moreover,

it is assumed that $g(\mathcal{X}) = \mathbf{w}^\top \mathcal{X}$ (i.e. the latent variable function is linear). This is visualized in Figure 12.1(a).

The (cumulative) probability that the label $\mathcal{Y}$ of an instance with a given feature vector $\mathcal{X} = \mathbf{x}$ is smaller than or equal to C_k is denoted $\Pr(\mathcal{Y} \leq C_k \mid \mathcal{X} = \mathbf{x})$. We now have that

$$\Pr(\mathcal{Y} \leq C_k \mid \mathcal{X} = \mathbf{x}) = \Pr(g(\mathbf{x}) + \mathcal{E} \leq \theta_k) = \Pr(\mathcal{E} \leq \theta_k - g(\mathbf{x}))\,.$$

Using the logistic distribution of $\mathcal{E}$ and $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, it follows that

$$\Pr(\mathcal{Y} \leq C_k \mid \mathcal{X} = \mathbf{x}) = \begin{cases} \dfrac{\exp(-\mathbf{w}^\top \mathbf{x} + \theta_k)}{1 + \exp(-\mathbf{w}^\top \mathbf{x} + \theta_k)} & , \text{ if } k = 1, \ldots, q-1\,, \\ 1 & , \text{ if } k = q\,, \end{cases}$$

This is visualized in Figure 12.1(b).

Fitting proportional odds models

We now use the short-hand notation $\overline{p}_k(\mathbf{x}) = \Pr(\mathcal{Y} \leq C_k \mid \mathcal{X} = \mathbf{x})$. The parameters $\mathbf{w}$ and $\theta_1, \ldots, \theta_{q-1}$ are obtained by maximum likelihood estimation. Given a dataset $(\mathbf{X}, \mathbf{Y})$ the likelihood is

$$\prod_{i=1}^{n} \Pr(\mathcal{Y} = \mathbf{y}_i \mid \mathcal{X} = \mathbf{x}_i)\,.$$

To simplify this expression, we use a recoding of the labels. Let $\mathbf{y}_i^* \in \{0,1\}^q$ such that $y_{i,k}^* = 1$ if $\mathbf{y}_i = C_k$ and $y_{i,k}^* = 0$ otherwise. Using the short-hand notation given before, we obtain

$$\prod_{i=1}^{n} \Pr(\mathcal{Y} = \mathbf{y}_i \mid \mathcal{X} = \mathbf{x}_i) = \prod_{i=1}^{n} \prod_{k=1}^{q-1} (\bar{p}_k(\mathbf{x}_i) - \bar{p}_{k+1}(\mathbf{x}_i))^{y_{i,k}^*}\,. \tag{12.5}$$

As a final step, we can take the negative logarithm of the likelihood function, and minimize the resulting expression w.r.t. $\mathbf{w}$ and $\theta_1, \ldots, \theta_{q-1}$.

12.3.2. Performance measures for ordinal regression problems

Besides model structure, another important difference between multi-class classification and ordinal regression can be found in the performance measure (loss function) that is used (optimized). To evaluate the performance of a given multi-class classification model, the accuracy on a test dataset is typically measured, and a differentiable approximation of accuracy such as the binomial deviance or hinge

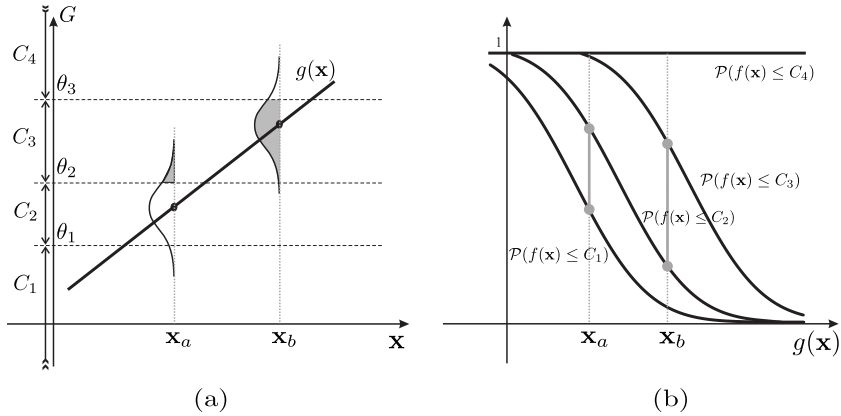


Figure 12.1: (a) A model underlying ordinal data in a 4-class case, the horizontal axis indicates the value of a (one-dimensional) feature vector $\mathbf{x}$, the vertical axis contains the value of the latent variable. The latent (random) variable $\mathcal{G}$ for a given feature vector $\mathcal{X} = \mathbf{x}$ is assumed to follow a logistic distribution with mean $g(\mathbf{x})$. θ_1, θ_2 and θ_3 represent the thresholds on the latent variable. The areas marked in gray indicate the probabilities $\Pr(f(\mathbf{x}_a) = C_3)$ and $\Pr(f(\mathbf{x}_b) = C_3)$. (b) A visualization of the proportional odds model with the latent variable on the horizontal axis and the cumulative class probabilities on the vertical axis.

loss is typically optimized on training data to fit the parameters of the model. In an ordinal setting, however, the use of accuracy seems unnatural. For instance, when the class labels are {*bad*, *moderate*, *good*}, classifying a *good* instance as *bad* is worse than classifying it as *moderate*. Accuracy does not take this into account. Specific performance measures should be used instead, such as the C-index or concordance index [163] or the volume under the ROC surface [164].

12.3.3. Modeling compositional outputs with ordered components

In multivariate regression analysis, the following (probabilistic) model is often assumed to underlay the data

$$\mathcal{Y} = \mathbf{f}(\mathcal{X}) + \mathcal{E} \tag{12.6}$$

with $\mathbf{f} : X \to \mathbb{R}^q$ and $\mathcal{E}$ a random vector of noise terms with $\mathbb{E}[\mathcal{E}] = \mathbf{0}_q$. Naturally, for a probabilistic model for compositional outputs, it is required that the codomain of $\mathbf{f}$ is $\mathbb{S}^q$. Moreover, the sample space of $\mathcal{Y}$ is $\mathbb{S}^q$.

Ordinal regression can be seen as a specific example of multi-class classification. Likewise, compositional data with ordered components are a specific example of compositional data. As a result, model (12.6) can be used in this setting as well.

However, it does not take the order on the classes into account. Therefore, it is shown that the ordinal information on the classes can be incorporated into (12.6) by adding an additional constraint on $\mathbf{f}$. Consider the following probabilistic model of compositional data

$$\mathcal{Y} = \mathbf{h}(g(\mathcal{X})) + \mathcal{E} \tag{12.7}$$

with $g : X \to \mathbb{R}$, $\mathbf{h} = (h_1, \ldots, h_q) : \mathbb{R} \to \mathbb{S}^q$, $\mathcal{E}$ is a random vector of noise terms and the sample space of $\mathcal{Y}$ is $\mathbb{S}^q$. It can easily be seen that (12.7) is a special case of (12.6). In the following paragraphs, it is shown that this model is particularly useful to model compositional data with ordered components.

An example

For illustrative purposes, we represent model (12.7) graphically for an example where the output space is $\mathbb{S}^3$ and the input space $\mathbb{R}$. Figure 12.2 gives a visual representation of the conditional probability density function $\rho_{\mathcal{Y}|\mathcal{X}}(. \mid \mathbf{x})$ of $\mathcal{Y}$ given $\mathcal{X} = \mathbf{x}$. The solid line represents $\mathbb{E}[\mathcal{Y} \mid \mathcal{X} = \mathbf{x}]$. The conditional random variable $\mathcal{Y}$ given $\mathcal{X} = \mathbf{x}$ is Dirichlet distributed with a fixed concentration parameter.

The dominance relation

In an ordinal regression setting, the latent variable motivation suggests that a monotone relationship exists between the latent variable and the output (the predicted class label). When modeling compositional data with ordered components, we want to preserve this monotone relationship between the latent variable and the output (which is in this case a composition). However, to be able to speak of a monotone relationship between a latent variable and a composition, an order relation has to be defined on the outputs. Unfortunately, unlike the crisp ordinal regression case, the general problem setting does not define a linear order on the compositions. Therefore, an order has to be assumed that reflects the natural ordering on the class labels. We propose an order relation that is strongly related to the concept of first order stochastic dominance [165], which defines a partial order relation on probability density functions. The concept of stochastic dominance was introduced in decision theory and it has for example been used in rough sets [166, 167] and instance-based learning algorithms [168, 169]. First, the notion of a cumulative q-part composition is introduced.

Definition 12.1. *For the q-part compositions $\mathbf{y}$ and $\mathbf{f}(\mathbf{x})$, the cumulative q-part compositions $\overline{\mathbf{y}} = (\overline{y}_1, \ldots, \overline{y}_q)$ and $\overline{\mathbf{f}}(\mathbf{x}) = (\overline{f}_1(\mathbf{x}), \ldots, \overline{f}_q(\mathbf{x}))$ are defined as*

$$\overline{y}_k = \sum_{l=1}^{k} y_l \quad \textit{and} \quad \overline{f}_k(\mathbf{x}) = \sum_{l=1}^{k} f_l(\mathbf{x}) \quad \textit{for} \quad k = 1, \ldots, q\,.$$

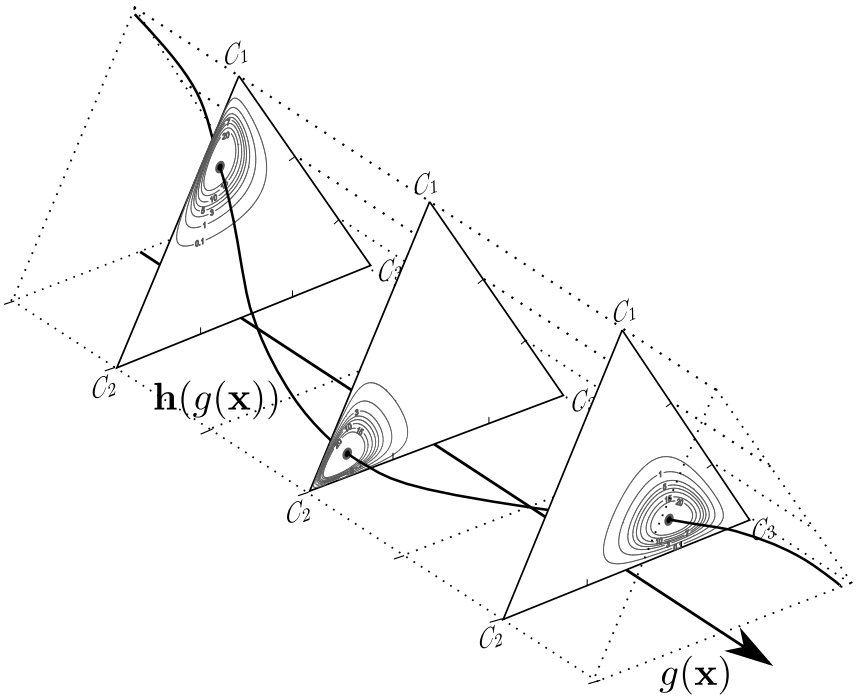


Figure 12.2: Visual representation of probabilistic model (12.7) with output space $\mathbb{S}^3$. The conditional probability density function $\rho_{\mathcal{Y}|\mathcal{X}}(. \mid \mathbf{x})$ of $\mathcal{Y}$ given $\mathcal{X} = \mathbf{x}$ is shown by means of several contour plots. The solid line represents $\mathbb{E}[\mathcal{Y} \mid \mathcal{X} = \mathbf{x}]$. Moreover, the conditional random variable $\mathcal{Y}$ given $\mathcal{X} = \mathbf{x}$ is Dirichlet distributed with a fixed concentration parameter.

Using cumulative compositions, the notion of stochastic dominance can be used to obtain a partial order relation on compositions, notwithstanding the fact that these vectors do not represent distributions.

Definition 12.2. *Given two q-part compositions $\mathbf{y}_1$, $\mathbf{y}_2 \in \mathbb{S}^q$, we say that $\mathbf{y}_1$ dominates $\mathbf{y}_2$ (denoted $\mathbf{y}_1 \succcurlyeq_{SD} \mathbf{y}_2$) if the following holds for the cumulative compositions $\overline{\mathbf{y}}_1 = (\overline{y}_{1,1}, \ldots, \overline{y}_{1,q})$ and $\overline{\mathbf{y}}_2 = (\overline{y}_{2,1}, \ldots, \overline{y}_{2,q})$:*

$$(\forall k \in \{1, \ldots, q\}) \left(\overline{y}_{1,k} \leq \overline{y}_{2,k}\right) .$$

We say that $\mathbf{y}_1$ strictly dominates $\mathbf{y}_2$ if

$$\mathbf{y}_1 \succcurlyeq_{SD} \mathbf{y}_2 \quad \textit{and} \quad \mathbf{y}_1 \neq \mathbf{y}_2 . \tag{12.8}$$

It can easily be seen that the dominance relation defines a partial order relation on a given set of q-part compositions. Moreover, it forms a generalization of the linear order on the label set. To illustrate this, consider two crisp labels $C_k, C_l \in C$. If $C_k \prec C_l$, the compositional representation of C_l will dominate the one of C_k. This dominance relation is now used to define the following property.

Definition 12.3. *A mapping $\mathbf{h} : \mathbb{R} \to \mathbb{S}^q$ is monotone with respect to the dominance relation if for any $s_1, s_2 \in \mathbb{R}$ the following implication holds:*

$$s_1 \geq s_2 \Rightarrow \mathbf{h}(s_1) \succcurlyeq_{SD} \mathbf{h}(s_2) .$$

Motivation of the ordinal model

The dominance relation, combined with the latent variable interpretation, can be used to motivate the structure of model (12.7). For this model, the vector of functions $\mathbf{f} : X \to \mathbb{S}^q$ was redefined as

$$\mathbf{f}(\mathbf{x}) = \mathbf{h}(g(\mathbf{x})) , \tag{12.9}$$

where $g : X \to \mathbb{R}$ and $\mathbf{h} = (h_1, ..., h_q) : \mathbb{R} \to \mathbb{S}^q$. In this form, $\mathbf{f}$ can be seen as a two-step process. Firstly, g determines an object's value for the latent variable. Secondly, this value is mapped to $\mathbb{S}^q$ by $\mathbf{h}$. Here, $\mathbf{h}$ can be chosen to be monotone with respect to the dominance relation, however, this requires some additional constraints on $\mathbf{h}$. Consider the cumulative counterpart $\overline{\mathbf{h}} = (\overline{h}_1, \ldots, \overline{h}_q)$ of $\mathbf{h}$, where

$$\overline{h}_k(u) = \sum_{l=1}^{k} h_l(u) \quad , \text{ for } k = 1, \ldots, q; \quad \forall u \in \mathbb{R}.$$

Note that a one-to-one correspondence exists between $\overline{\mathbf{h}}$ and $\mathbf{h}$. This duality is exploited in the following obvious proposition, which states that requiring $\overline{h}_1, \ldots, \overline{h}_{q-1}$

to be decreasing functions suffices to obtain a monotone mapping.

Proposition 12.1. *A mapping* $\mathbf{h} : \mathbb{R} \to \mathbb{S}^q$ *is monotone with respect to the dominance relation if and only if* $\overline{h}_1, \ldots, \overline{h}_{q-1}$ *are decreasing functions.*

The constraints on $\overline{h}_1, \ldots, \overline{h}_{q-1}$ are illustrated in Figure 12.3. They assure that an object will dominate all objects having a smaller value for the latent variable.

The discussion above suggests that, with a proper choice of $\mathbf{h}$, model (12.7) can be used to model compositional data with ordered components. In particular, when $\mathbf{h}$ is monotone with respect to the dominance relation, an increase in the value of the latent variable will lead to an increase in the porportional contribution of the higher classes. These findings are summarized in the following property.

Definition 12.4. *Model (12.7) is called monotone with respect to the dominance relation if for any two feature vectors* $\mathbf{x}_1$ *and* $\mathbf{x}_2$*, the following equivalence holds*

$$\mathbb{E}[\mathcal{Y} \mid \mathcal{X} = \mathbf{x}_1] \succcurlyeq_{SD} \mathbb{E}[\mathcal{Y} \mid \mathcal{X} = \mathbf{x}_2] \Leftrightarrow g(\mathbf{x}_1) \geq g(\mathbf{x}_2)\,,$$

where $\mathbb{E}[\mathcal{Y} \mid \mathcal{X} = \mathbf{x}_i]$ *represents the expected value of label* $\mathcal{Y}$*, conditioned on the input* $\mathcal{X} = \mathbf{x}_i$*.*

We now provide sufficient conditions such that probabilistic model (12.7) is monotone with respect to the dominance relation. These conditions are provided by means of the following proposition, of which the proof is a trivial consequence of the discussion above and the properties of the Dirichlet distribution given in Chapter 2 .

Proposition 12.2. *Model (12.7) is monotone with respect to the dominance relation if the following properties hold:*

(i) $\overline{h}_1, \ldots, \overline{h}_{q-1}$ *are decreasing functions*

(ii) The conditional distribution of $\mathcal{Y}$ *given* $\mathcal{X} = \mathbf{x}_i$ *is a Dirichlet distribution with parameter set* $\boldsymbol{\beta}(\mathbf{x}) = (\beta_1(\mathbf{x}), \ldots, \beta_K(\mathbf{x}))$ *where*

$$\beta_k(\mathbf{x}) = s\, h_k(g(\mathbf{x})) \quad , \text{ for } k = 1, \ldots, q; \quad \mathbf{x} \in X$$

where $s \in \mathbb{R}_0^+$*.*

12.3.4. Performance measures for compositions with ordered components

When modeling compositional data without an ordering on the components, the Euclidean or Manhattan distance between a prediction/label pair $(\mathbf{f}(\mathbf{x}), \mathbf{y})$ can be used as loss function. However, such a loss function is not suitable in case of

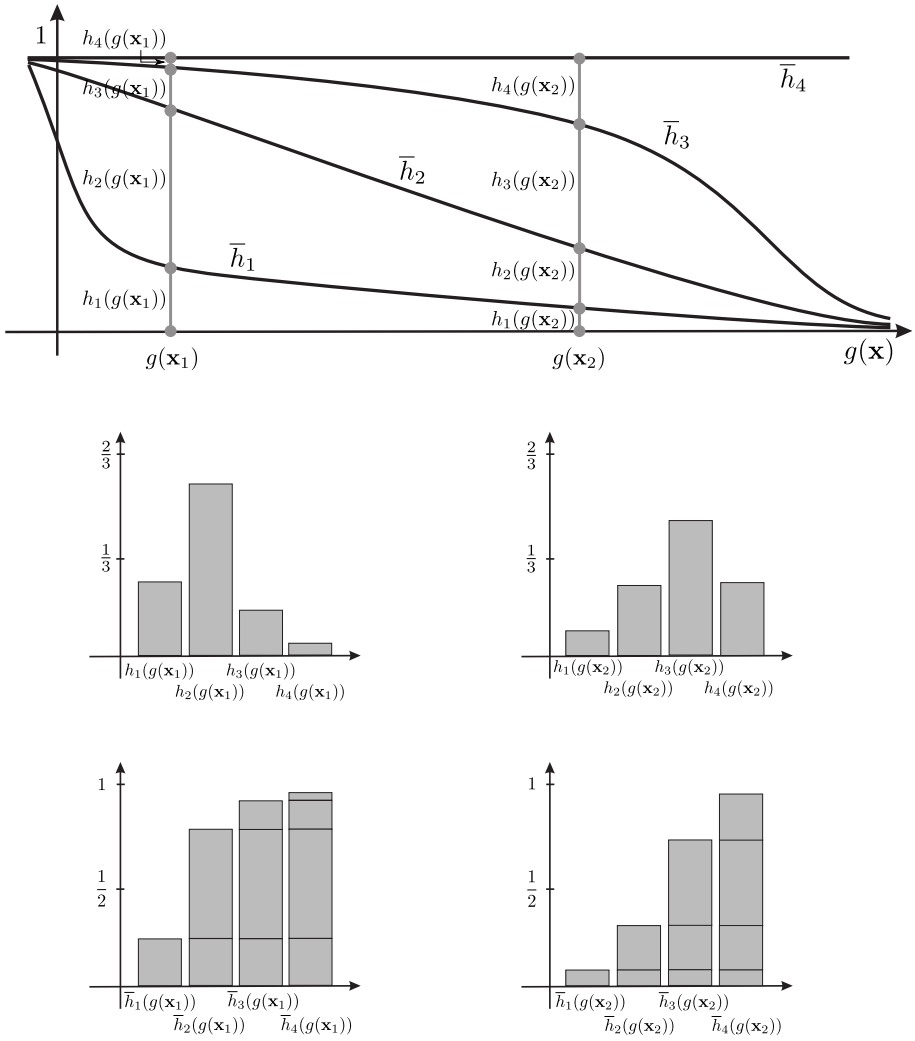


Figure 12.3: (Top) For 4-part compositions with ordered components, an example of a set $\{\overline{h}_1, \ldots, \overline{h}_4\}$ is given that establishes a monotone relationship between the latent variable and the resulting compositions. (Middle) Two compositions constructed from these functions. (Bottom) The cumulative counterparts of the compositions in the middle.

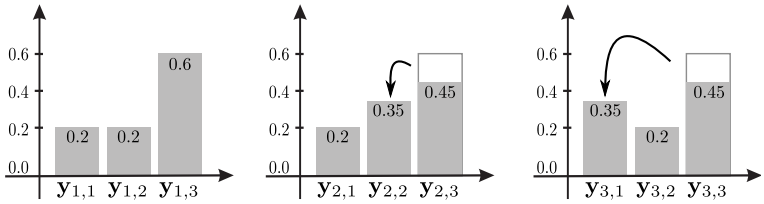


Figure 12.4: Illustration of three compositions with $q = 3$. $\mathbf{y}_1$ and $\mathbf{y}_2$ differ through an exchange between the proportions of C_1 and C_2. $\mathbf{y}_1$ and $\mathbf{y}_3$ differ through the same amount of exchange between C_1 and C_3.

compositional data with ordered components since it neglects the order on the classes. To illustrate this, consider the compositions $\mathbf{y}_1$, $\mathbf{y}_2$ and $\mathbf{y}_3$ in Figure 12.4. In this case, the Manhattan distance between $\mathbf{y}_1$ and $\mathbf{y}_2$ equals the Manhattan distance between $\mathbf{y}_1$ and $\mathbf{y}_3$. When an ordering $C_1 \prec C_2 \prec C_3$ is present, this seems unnatural: $\mathbf{y}_2$ differs from $\mathbf{y}_1$ through an exchange between the proportions of C_2 and C_3. On the other hand, $\mathbf{y}_3$ and $\mathbf{y}_1$ differ through the same amount of exchange, but now between C_1 and C_3, which are further apart. To incorporate the order into the performance measure, the Manhattan distance on the cumulative compositions can be used instead [170, 171]. Formally, we call this the mean absolute error on the cumulative compositions (ℓ_{MAEC}):

$$\ell_{\text{MAEC}}(\mathbf{f}(\mathbf{x}), \mathbf{y}) = \sum_{k=1}^{q} \left| \overline{f}_k(\mathbf{x}) - \overline{y}_k \right| . \tag{12.10}$$

Applying this to the example in Figure 12.4 gives $\ell_{\text{MAEC}}(\mathbf{y}_1, \mathbf{y}_2) < \ell_{\text{MAEC}}(\mathbf{y}_1, \mathbf{y}_3)$.

12.3.5. A proportional odds model for predictive modeling of compositional outputs with ordered components

The proportional odds model as introduced in Section 12.3.1 naturally establishes a monotone relationship between the latent variable and the estimated class probabilities, since the logistic functions fitted by this model on the latent variable axis respect the constraints given in Proposition 12.1 (see for instance Figure 12.1(b)). As a result, the fitted logistic curves are particularly useful to model data with an underlying model of type (12.7). Based on these findings, we propose a minor extension of the proportional odds model for learning to predict compositions with ordered responses. As a hypothesis space H, we consider the set of vector functions $\mathbf{f} : X \to \mathbb{S}^q$ of which the cumulative counterpart can be written as follows:

$$\overline{f}_k(\mathbf{x}) = \begin{cases} \dfrac{\exp(-\mathbf{w}^\top \mathbf{x} + \theta_k)}{1 + \exp(-\mathbf{w}^\top \mathbf{x} + \theta_k)} & \text{, if } k = 1, \ldots, q-1 \,, \\ 1 & \text{, if } k = q \,, \end{cases} \tag{12.11}$$

with $\mathbf{w} \in \mathbb{R}^p$ and $\theta_1 \leq \ldots \leq \theta_{q-1} \in \mathbb{R}$.

This construction naturally implies that each function $\mathbf{f} \in H$ can be written as $\mathbf{h}(g(\mathbf{x}))$ where $\mathbf{h}$ is monotone with respect to the dominance relation and $g(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$.

To fit the traditional proportional odds model to a dataset, the negative log-likelihood (see Eq. (12.5)) is usually considered as a loss function. In our approach, we present a slightly modified loss function that also takes a regularization term into account:

$$r(\mathbf{f}) = \lambda \|\mathbf{w}\|_2^2 - \log\left(\prod_{i=1}^{n}\prod_{k=1}^{q-1}(\bar{f}_k(\mathbf{x}_i) - \bar{f}_{k+1}(\mathbf{x}_i))^{y_{i,k}}\right) \tag{12.12}$$

$$= \lambda \|\mathbf{w}\|^2 - \sum_{i=1}^{n}\log\Big(\overline{f}_1(\mathbf{x}_i)^{y_{i,1}}\Big) - \sum_{i=1}^{n}\sum_{k=2}^{q}\log\Big((\overline{f}_k(\mathbf{x}_i) - \overline{f}_{k-1}(\mathbf{x}_i))^{y_{i,k}}\Big), \tag{12.13}$$

In this way, the original likelihood function, which has a completely probabilistic interpretation, is here adopted as a regular loss function. As such, this loss function looses its probabilistic interpretation, because compositions should not necessarily be interpreted as prior probability estimates. However, the above loss function has interesting properties, since it is differentiable and convex, leading to an optimization problem that retains its computational tractability. The parameters $\mathbf{w}$ and $\theta_1, \ldots, \theta_{q-1}$ can be estimated with a gradient descent algorithm.

As the proportional odds model for predictive modeling of compositional data with ordered responses can be seen as a special case of the setting described in Section 12.2, it can be kernelized as well. Given a kernel $\kappa : X \times X \to \mathbb{R}$, a reproducing kernel Hilbert space H of functions can be constructed (following the methodology of Chapter 11). Following [133], the minimizer $\hat{\mathbf{f}}$ of the regularized empirical loss function

$$r(\mathbf{f}) = \lambda \|g\|_H^2 - \sum_{i=1}^{n}\log\Big(\overline{f}_1(\mathbf{x}_i)^{y_{i,1}}\Big) - \sum_{i=1}^{n}\sum_{k=2}^{q}\log\Big(\big(\overline{f}_k(\mathbf{x}_i) - \overline{f}_{k-1}(\mathbf{x}_i)\big)^{y_{i,k}}\Big),$$

admits a representation of the form $\hat{g}(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i)$ (along with the intercepts $\hat{\theta}_i$, $i = 1, \ldots, q-1$). Therefore, the regularization term can be written as $\|g\|_H^2 = \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha}$, where $\mathbf{K}$ is the Gram matrix, *i.e.* $\mathbf{K}_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. Similar to the standard proportional odds model, the model parameters can be estimated by minimizing the (regularized) loss function. Since the loss function retains its convexity and differentiability, standard optimization algorithms such as gradient descent can be applied to this end. The kernel form of the regularized empirical loss function was implemented in R [172]. To minimize this loss function the BFGS

algorithm (which is a quasi-Newton method) [173] that is included in the base distribution of R was used.

Remark. The proportional odds models requires that $\hat{\theta}_1 \leq \hat{\theta}_2 \leq \ldots \leq \hat{\theta}_{q-1}$. Therefore, it could be argued that these inequalities should be implemented as constraints in the resulting optimization problem. Unfortunately, standard implementations of the Broyden-Fletcher-Goldfarb-Shanno algorithm (BFGS) do not allow constraints to be included. However, it is easy to see that at the minimizer of r, the inequality constraints will generally not be active. Therefore, given a feasible starting point of the BFGS algorithm, the iterates will generally not leave the feasible part of the search space. In the unlikely case that an iterate would leave the feasible region, this can be remedied by reducing the step size for that iteration.

12.4. Experiments on synthetic data

When the set of class labels is equipped with an order, it should be beneficial (in terms of predictive performance) to use models that take this ordering into account. To verify this claim in case of compositional data with ordered components, a series of experiments on synthetic data is presented in this section. On these datasets, the performance of the kernelized proportional odds model for compositional outputs with ordered components (POC) will be compared with some alternative (less model-driven) approaches. It should be noted that, to the best of our knowledge, there are no existing models capable of learning compositional data with ordered components.

12.4.1. Alternative methods

For a given training dataset T of size n, the mean[1] composition $\mathbf{y}_T^*$ can be obtained as

$$\mathbf{y}_T^* = \frac{1}{n}\left(\sum_{(\mathbf{x}_i,\mathbf{y}_i)\in T} y_{i,1}, \ldots, \sum_{(\mathbf{x}_i,\mathbf{y}_i)\in T} y_{i,q}\right). \tag{12.14}$$

As a baseline method, this mean can be used as a model, $\mathbf{f}(\mathbf{x}) = \mathbf{y}_T^*$ (referred to as Mean). Secondly, as a first 'real' model, we choose to modify random forest regression (RF) as introduced in [124], to enable it to learn compositional data with ordered components. Since the standard entropy measure (mean squared error) has no meaning in this setting, a different entropy measure is used. For a given node and associated dataset T_l, the average value of $\ell_{\text{MAEC}}(\mathbf{y}_{T_l}^*, \mathbf{y})$, where $\mathbf{y}$ runs over all

[1] Note that we use the traditional arithmetic mean here, as opposed to the average that is implied by the Aitchison geometry.

compositions in T_l, is used as a entropy measure. This modified entropy measure assures that the random forest directly optimizes ℓ_{MAEC}. Similar to the original algorithm, among all possible splits, the candidate that maximizes the decrease in entropy is chosen to expand the tree. As with regression trees, an ensemble of trees is grown on bootstrap samples of the training set and the predictions are averaged to obtain the final result. However, since the predictions are compositions instead of real numbers, the average is taken over these compositions.

As a third alternative, we also compare with the extended version of multi-class kernel logistic regression model for compositions with unordered components, as proposed in [174] by extending multi-class kernel logistic regression (MKLR). The extended MKLR algorithm is not specifically designed for an ordinal setting, but it delivers quite satisfactory results for learning from compositional data when no order is present on the classes. Since it can be seen as a variant of the proportional odds model without the latent variable assumption, it seems obvious to compare our extension of the proportional odds model with this algorithm. Moreover, standard kernel logistic regression is one of the state-of-the-art algorithms in multi-class classification.

12.4.2. Experimental setup

With some carefully designed experiments on synthetic data, we will try to demonstrate the performance of our algorithms and illustrate some specific properties. In all experiments, model (12.7) is used as the model underlying the data. More precisely, we will investigate the influence of g and $\mathbf{h}$ on the predictive capabilities of the models. Both g and $\mathbf{h}$ strongly influence the complexity of the model. The latent variable function g can vary from a (simple) linear to a complex, highly nonlinear function. On the other hand, due to the constraints in Proposition 12.1, the functions $h_1, \ldots, h_4$ are highly nonlinear by definition, making it difficult to speak in terms of complexity for these functions. However, as mentioned in Section 12.3.5, the POC constructs $\overline{\mathbf{h}}$ by means of a set of logistic functions. Consequently, a vector of functions $\overline{\mathbf{h}}$ that cannot be approximated well by means of logistic functions might require a more complex model.

The (joint) influence of g and $\mathbf{h}$ on the predictive performance will be investigated in four different experimental settings (A–D) (Table 12.1). In Setting A, g is linear and $\mathbf{h}$ is a set of logistic functions (Figure 12.5(a)). In Setting B, g is linear and $\mathbf{h}$ strongly deviates from the logistic shape (Figure 12.5(b)). In Setting C, g is nonlinear and $\mathbf{h}$ strongly deviates from the logistic shape (Figure 12.5(b)). In Setting D, g is nonlinear and $\mathbf{h}$ constitutes a typical Ruspini partition (Figures 12.5(c) and 12.5(d)). In this last experiment, we want to simulate a setting in which the functions h_i have a traditional fuzzy set-like shape.

Table 12.1: Overview of experimental settings A-D. In the linear case, $g(\mathbf{x}) = 4(x_1 + x_2 + x_3 + x_4) + x_5 + x_6 + 2x_7 - 25$. In the nonlinear case, $g(\mathbf{x}) = 9\sqrt{\left|\frac{x_1+x_2-x_3-x_4}{x_5+x_6-x_7}\right|} - 1.2$.

	$\overline{\mathbf{h}}$ logistic (Figure 12.5(a))	$\overline{\mathbf{h}}$ non-logistic (Figure 12.5(b))	$\mathbf{h}$ Ruspini partition (Figure 12.5(c))
g linear	Setting A	Setting B	-
g nonlinear	-	Setting C	Setting D

12.4.3. Data generation and parameter estimation

For each of Settings A-D, a synthetic dataset of size n can be generated using the following scheme:

1. Choose an input space X and define a random vector $\mathcal{X}$ with probability density function $f_{\mathcal{X}}$, $g : X \to \mathbb{R}$, $\mathbf{h} : \mathbb{R} \to \mathbb{S}^4$ monotone with respect to the dominance relation (e.g. Table 12.1).

2. To obtain a set of n feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$:

   (i) Choose an interval $[a, b] \subset \text{range}(g)$ and divide it in n non-overlapping sub-intervals of equal length.

   (ii) Sample candidate feature vectors $\mathbf{x}$ from $f_{\mathcal{X}}$ until sufficient values of each sub-interval of $g(\mathbf{x})$ are retrieved so that each subinterval contains at least one.

   (iii) From each sub-interval, randomly choose one feature vector to obtain $\mathbf{x}_1, \ldots, \mathbf{x}_n$.

3. For each selected feature vector $\mathbf{x}_i$, obtain $\mathbf{y}_i$ as a sample from

$$\text{Dir}(s\, h_1(g(\mathbf{x}_i)), \ldots, s\, h_4(g(\mathbf{x}_i)))\,.$$

   with $s \in \mathbb{R}^+$ a parameter.

According to Proposition 12.2, the obtained dataset originates from a model that is monotone with respect to the dominance relation. As such, this data generation process represents a setting with compositional outputs with ordered components. Note that, instead of step 2, the feature vectors could have been sampled directly from $f_{\mathcal{X}}$. However, the procedure described here ensures that the values for $g(\mathbf{x})$ are uniformly spread over a chosen interval. With an appropriate choice for this interval w.r.t. $\mathbf{h}$, this increases the variability of $\mathbf{h}(g(\mathbf{x}_1)), \ldots, \mathbf{h}(g(\mathbf{x}_n))$ in the dataset, leading to more interesting cases.

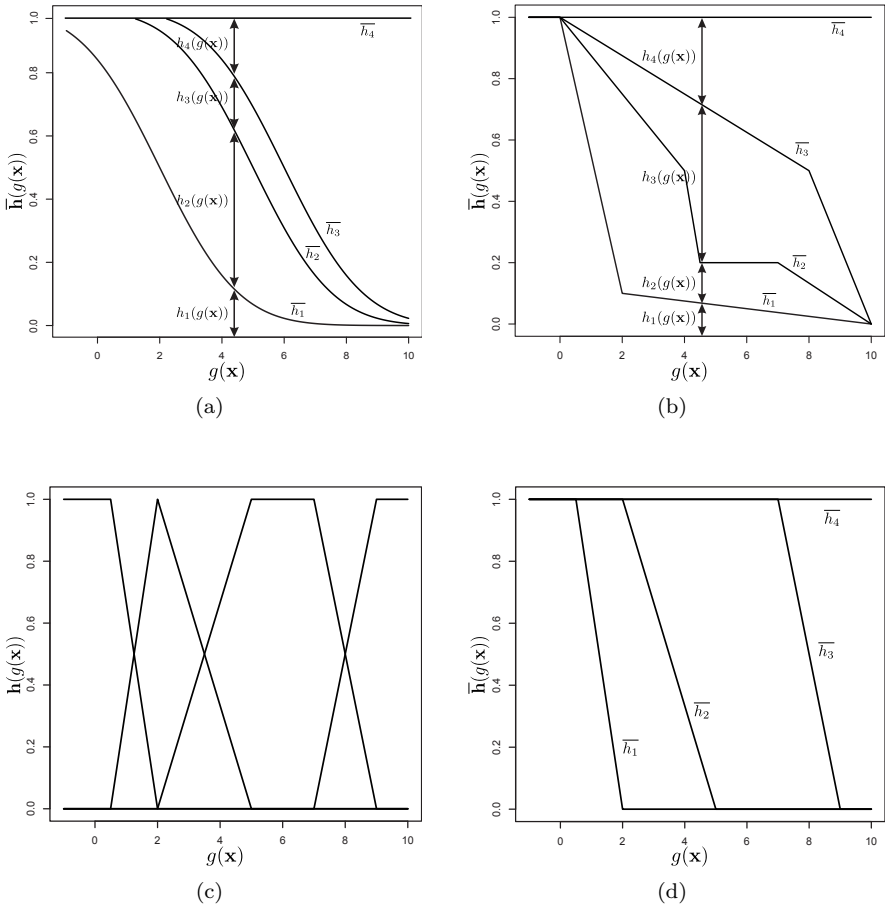


Figure 12.5: (a) Representation of $\overline{\mathbf{h}}$ in Setting A. (b) Representation of $\overline{\mathbf{h}}$ in Settings B and C. (c) $h_1, \ldots, h_4$ constitute a typical Ruspini partition, this partition is monotone with respect to the dominance relation and used in Setting D. (d) The cumulative version of (c).

For each setting, a training set containing 20 data points was created. $f_{\mathcal{X}}$ was chosen uniform over $[1,7]^7$, $[a,b] = [-1,10]$ and $s = 100$. A separate validation set is sampled to tune the hyper-parameters (regularization parameter λ and kernel parameter σ), with a grid search from 2^{-8} to 2^8 and a step size 2 (multiplicative) for both parameters. Besides the extended proportional odds model and the MKLR model, we also tune the enhanced random forest classifier that serves as a comparison. Based on the out-of-bag error, the number of trees is set to 500 and the number of candidate features for each split is set to 3. The final performance statistics are based on a test set of size 1000. Each experiment is repeated 30 times. For each of these settings, the performance (in terms of $\mathcal{L}_{\text{MAEC}}$) is calculated for 6 different methods: global mean (Mean), random forest (RF), MKLR with linear kernel (MKLR-Lin), MKLR with RBF kernel (MKLR-RBF), proportional odds for compositions with a linear kernel (POC-Lin) and proportional odds for compositional outputs with an RBF kernel (POC-RBF). Note that the use of a linear kernel is the same as using no basis expansion, resulting in a linear model for the latent variable.

12.4.4. Results

Table 12.2 summarizes the performance (in terms of the average ℓ_{MAEC}) for all 6 methods in each setting, averaged over 30 repetitions. In setting A, POC substantially outperforms the other methods. This does not come as a surprise, because Setting A was especially designed to favor our approach (g linear and $\overline{\mathbf{h}}$ logistic). The use of an RBF kernel does not increase performance since linear models are flexible enough for this setting. Furthermore, one can also observe that the RF classifier behaves much worse than the kernel methods. This phenomenon can be explained by the fact that we are fitting here a linear model, which can be easily simulated with a linear kernel, while random forests have more difficulties with simulating such models.

For Setting B, it can be seen that the main conclusions from Setting A remain valid when the components of $\overline{\mathbf{h}}$ deviate from the logistic functions, but the use of an RBF kernel now seems to improve performance. This observation supports the claim that, notwithstanding the linearity of g, more flexible models are able to improve performance when the components of $\overline{\mathbf{h}}$ are not logistic. Thus, although the kernelized proportional odds model considers a specific logistic shape for $\overline{\mathbf{h}}$, it can fit datasets with other shapes quite well. In Setting C, the need for nonlinear models increases. Two main conclusions can be drawn from this experiment, both of them being in line with initial expectations. Firstly, as the complexity of the latent variable function increases, the performance in terms of ℓ_{MAEC} decreases. This observation holds for all models that are investigated. Secondly, the need for using nonlinear models becomes even more striking than in Setting B. POC with an RBF kernel still obtains the best performance, followed by MKLR with an

Table 12.2: Summarizing table of the experiments on synthetic data for the four different settings. For each method, the average ℓ_{MAEC} over 30 repetitions is given and the standard deviation is shown between brackets.

Method	Setting A	Setting B	Setting C	Setting D
Mean	0.924 (0.020)	0.686 (0.032)	0.645 (0.040)	0.888 (0.050)
RF	0.566 (0.050)	0.435 (0.032)	0.583 (0.042)	0.798 (0.045)
MKLR-Lin	0.123 (0.017)	0.280 (0.035)	0.601 (0.048)	0.800 (0.090)
MKLR-RBF	0.145 (0.024)	0.267 (0.030)	0.514 (0.064)	0.680 (0.076)
POC-Lin	0.108 (0.012)	0.242 (0.016)	0.574 (0.049)	0.751 (0.067)
POC-RBF	0.107 (0.013)	0.232 (0.016)	0.479 (0.050)	0.551 (0.072)

RBF kernel, but their linear counterparts cannot predict the data well, since the performance of the RF classifier is now similar to the kernel methods with linear kernels. So, one can conclude that POC remains a useful tool for datasets with nonlinear relationships, but definitely nonlinear kernel functions have to be used. Finally, consider Setting D. Here as well, POC-RBF clearly outperforms all other methods. The same reasoning as in Setting B can be adopted. The kernel function is able to correct again for the triangular or trapezoidal shape of $\mathbf{h}$.

12.5. An illustration in agriculture

In this section we return to the introductory motivating example, where the goal consists of learning a predictive model that can handle compositional data with ordered components.

We explored a dataset presented in [152]. This dataset describes the degree of infection of cereal crops with head blight disease in Flanders (Belgium). To describe the degree of infection of a wheat field, a number of ears are selected randomly from the field. Each ear is classified by an expert in one of five ordinal classes: not infected, slightly infected, moderately infected, heavily infected, fully infected. As such, the fractions of ears in each class constitute a composition describing the infection degree of the field. It is known that the infection degree of a field with head blight is strongly influenced by the micro-climate during the flowering season. In this scope, 45 micro-climatological variables were recorded at each field. In total, 210 fields were analyzed in the year 2007 and 248 were analyzed in 2008.

The climatological variables can be used to build a predictive model for the infection degree of a field. Accordingly, MKLR-Lin and POC-Lin were fit to this dataset. In a first experiment, the data of 2007 was used to train the model, subsequently the performance of this model was evaluated on the data of 2008. In the second experiment, the data of 2008 was used for training and the data of 2007 for

Table 12.3: Performance results of MKLR-Lin, and POC-Lin in terms of ℓ_{MAEC} on the head blight dataset.

Method	2007	2008
MKLR-Lin	0.79	0.20
POC-Lin	0.72	0.19

performance evaluation. Table 12.3 shows the performance of MKLR-Lin and POC-Lin in terms of ℓ_{MAEC}. The results indicate that POC is able to outperform MKLR in terms of predictive power. However, it should be noted that the difference between both methods is rather limited. Moreover, when plotting the predicted outputs versus the observed outputs, it can be seen that the output cannot be predicted accurately. More strongly, we concluded that there exists close to no relationship (in this dataset) between the inputs and the outputs. Unfortunately, the dataset turned out to be inadequate to make statements about the superiority of one of the two methods. Therefore, we do not elaborate further on this experiment.

12.6. Modeling compositional data with ordered components on a bounded domain

Recall from Section 12.3.5 that vector functions of the form $\mathbf{f}(\mathbf{x}) = \mathbf{h}(g(\mathbf{x}))$ are interesting candidates for modeling compositional data with ordered components, provided that $\mathbf{h}$ is monotone with respect to the dominance relation. The extension of the proportional odds model presented in Section 12.3 uses a class of functions $\mathbf{f}$ of which the cumulative counterpart can be written as

$$\overline{f}_k(\mathbf{x}) = \begin{cases} \dfrac{\exp(-\mathbf{w}^\top \mathbf{x} + \theta_k)}{1 + \exp(-\mathbf{w}^\top \mathbf{x} + \theta_k)} & \text{, if } k = 1, \ldots, q-1\,, \\ 1 & \text{, if } k = q\,. \end{cases} \tag{12.15}$$

This parametrization ensures that the implied vector function $\mathbf{h}$ is monotone with respect to the dominance relation. However, the hypothesis space implied by this parametrization can be too restrictive in several situations. Indeed, one can easily find a vector function $\mathbf{h}$ that is monotone with respect to the dominance relation, but cannot be written in the form implied by Eq. (12.15) (we provide several examples later in this section). Therefore, we propose a generalization of the proportional odds model that allows for a more flexible modeling of compositional data with ordered components. This generalization requires that the input domain X is a bounded subset of $\mathbb{R}^p$. Moreover, we will illustrate that the original proportional odds model can be seen as a special case for which $X = \mathbb{R}^p$.

12.6.1. The hypothesis space of the proportional odds model

As a starting point, we formally describe the hypothesis space that is implied by the proportional odds model. For the remainder of this section, we will assume that the input space X is a subset of $\mathbb{R}^p$. The output space $Y = \mathbb{S}^q$. Moreover, let $\mathbb{S}^{q^X} = \{\mathbf{f} : X \to \mathbb{S}^q\}$ be the space of functions that map X into $\mathbb{S}^q$. Let

$$\mathbb{R}^{q-1}_{\leq} = \left\{\boldsymbol{\theta} \in \mathbb{R}^{q-1} \mid (\forall k \in \{1, \ldots, q-2\})(\theta_k \leq \theta_{k+1})\right\} .$$

Now, consider the following mapping:

$$\begin{aligned} \phi : \mathbb{R}^p \times \mathbb{R}^{q-1} &\to \mathbb{S}^{q^X} \\ (\mathbf{w}, \boldsymbol{\theta}) &\mapsto \begin{pmatrix} \exp(-\langle \mathbf{w}, . \rangle + \theta_1)/(1 + \exp(-\langle \mathbf{w}, . \rangle + \theta_1)) \\ \exp(-\langle \mathbf{w}, . \rangle + \theta_2)/(1 + \exp(-\langle \mathbf{w}, . \rangle + \theta_2)) \\ \vdots \\ 1 \end{pmatrix} . \end{aligned} \tag{12.16}$$

Using the mapping ϕ, we can easily construct the cumulative counterpart of the hypothesis space of the proportional odds model:

$$\bar{H} = \{\phi(\mathbf{w}, \boldsymbol{\theta}) \mid (\mathbf{w}, \boldsymbol{\theta}) \in \mathbb{R}^p \times \mathbb{R}^{q-1}_{\leq}\} . \tag{12.17}$$

From this formulation, it is clear that for each $\bar{\mathbf{f}} \in \bar{H}$, the components $\bar{f}_1, \ldots, \bar{f}_{q-1}$ share the same parameter vector $\mathbf{w}$. This characteristic is a natural result of the latent variable motivation of the proportional odds model. However, in some cases, such an approach may be too restrictive. As an example, consider a phenomenon for which the (expected) cumulative compositions are modeled by the vector function $\bar{\mathbf{b}} : [0, 1] \to [0, 1]^3$:

$$\bar{\mathbf{b}}(x) = \begin{pmatrix} \exp(x)/(1 + \exp(x)) \\ \exp(a\,x + 1)/(1 + \exp(a\,x + 1)) \\ 1 \end{pmatrix} ,$$

where $a \geq 1$ is a parameter.

When $a = 1$ we have that $\bar{\mathbf{b}} \in \bar{H}$. However, when $a > 1$ it is clear that $\bar{\mathbf{b}} \notin \bar{H}$. Such cases can occur when the latent variable assumption is not fulfilled. To model such phenomena, three options exist:

(i) Use the proportional odds model to approximate $\bar{\mathbf{b}}$. This can lead to a good model when $a \approx 1$, *i.e.* the latent variable assumption is only mildly violated.

(ii) Use multi-class logistic regression to approximate $\bar{\mathbf{b}}$. This strategy can lead to a good model when $a >> 1$, *i.e.* the latent variable interpretation is heavily violated.

(iii) Slightly relax the latent variable assumption and use a set of functions that extends $\bar{H}$.

In the following section, we elaborate upon the third option.

12.6.2. An extended hypothesis space

The hypothesis space $\bar{H}$ can be extended to include functions such as $\bar{\mathbf{b}}$. However, we need to take into account that such an extended set still represents cumulative compositions. This means that the (extended) hypothesis space $\bar{G}$ should at least satisfy the following conditions.

1. For any $\bar{\mathbf{f}} \in \bar{G}$ and $\mathbf{x} \in X$, we should have that $\bar{f}_1(\mathbf{x}) \geq 0$.
2. For any $\bar{\mathbf{f}} \in \bar{G}$ and $\mathbf{x} \in X$, we should have that $\bar{f}_k(\mathbf{x}) \leq \bar{f}_{k+1}(\mathbf{x})$, for $k = 1, \ldots, q-1$.
3. For any $\bar{\mathbf{f}} \in \bar{G}$ and $\mathbf{x} \in X$, we should have that $\bar{f}_q(\mathbf{x}) = 1$.

Such a hypothesis space can be constructed using the approach that was used to define $\bar{H}$. Therefore, consider the mapping ψ:

$$\begin{aligned} \psi : \bigtimes_{k=1}^{q-1} \mathbb{R}^p \times \mathbb{R}^{q-1} &\to \mathbb{S}^{q\,X} \\ ((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) &\mapsto \begin{pmatrix} \exp(-\langle \mathbf{w}_1, . \rangle + \theta_1)/(1 + \exp(-\langle \mathbf{w}_1, . \rangle + \theta_1)) \\ \exp(-\langle \mathbf{w}_2, . \rangle + \theta_2)/(1 + \exp(-\langle \mathbf{w}_2, . \rangle + \theta_2)) \\ \vdots \\ 1 \end{pmatrix} . \end{aligned} \tag{12.18}$$

Similar to Eq. (12.17), we could construct a hypothesis space

$$\left\{ \psi((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid ((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \in \bigtimes_{k=1}^{q-1} \mathbb{R}^p \times \mathbb{R}^{q-1} \right\} .$$

Unfortunately, this space does not satisfy the second of the conditions given before. To resolve this issue, we can attempt to remove the elements that violate the second property. Mathematically, the removal of those elements boils down to reducing

the set above to a new set

$$\left\{\psi((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid ((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \in O\right\},$$

where O is a subset of $\bigtimes_{k=1}^{q-1} \mathbb{R}^p \times \mathbb{R}^{q-1}$. This reasoning does not state how to choose O. In the remainder of this section, we develop a characterization of O that is amenable for optimization purposes.

As a starting point, recall that the second property requires that for any $k \in \{1, \ldots, q-2\}$ and for any $\mathbf{x} \in X$ we have that

$$\frac{\exp(-\mathbf{w}_k^\top \mathbf{x} + \theta_k)}{1 + \exp(-\mathbf{w}_k^\top \mathbf{x} + \theta_k)} \leq \frac{\exp(-\mathbf{w}_{k+1}^\top \mathbf{x} + \theta_{k+1})}{1 + \exp(-\mathbf{w}_{k+1}^\top \mathbf{x} + \theta_{k+1})},$$

which is equivalent to

$$-\mathbf{w}_k^\top \mathbf{x} + \theta_k \leq -\mathbf{w}_{k+1}^\top \mathbf{x} + \theta_{k+1},$$

or also

$$(\mathbf{w}_{k+1} - \mathbf{w}_k)^\top \mathbf{x} + \theta_k - \theta_{k+1} \leq 0. \tag{12.19}$$

When $X = \mathbb{R}^p$, this condition implies that $\mathbf{w}_{k+1} = \mathbf{w}_k$ and $\theta_k \leq \theta_{k+1}$. However, this constraint can be relaxed by explicitly taking into account that X is a bounded subset of $\mathbb{R}^p$. In this case, a convex polytope B can be defined such that $X \subseteq B$. Formally, using an $m \times p$ matrix $\mathbf{B}$ and an m-vector $\mathbf{b}$, this polytope is defined as:

$$B = \{\mathbf{x} \in \mathbb{R}^p \mid \mathbf{B}\mathbf{x} \leq \mathbf{b}\}. \tag{12.20}$$

Using this polytope, the problem of finding a good hypothesis space reduces to the problem of choosing O such that for any $\mathbf{x} \in \mathbb{X}$ and for any $((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \in O$, it holds that

$$\bigwedge_{k=1}^{q-2} (\mathbf{w}_{k+1} - \mathbf{w}_k)^\top \mathbf{x} + (\theta_k - \theta_{k+1}) \leq 0, \tag{12.21}$$

where $\bigwedge_{k=1}^{q-2}$ denotes the Boolean conjunction of the $q-2$ clauses.

Let us now focus on one of these clauses, *i.e.*

$$(\mathbf{w}_{k+1} - \mathbf{w}_k)^\top \mathbf{x} + (\theta_k - \theta_{k+1}) \leq 0.$$

Notably, the requirement that inequality (12.19) holds for any $\mathbf{x} \in B$ is equivalent to requiring that the system

$$\begin{aligned} \mathbf{B}\,\mathbf{x} &\leq \mathbf{b} \\ (\mathbf{w}_{k+1} - \mathbf{w}_k)^\top \mathbf{x} + (\theta_k - \theta_{k+1}) &> 0 \end{aligned}$$

has no solution w.r.t. $\mathbf{x}$. The vectors $((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta})$ for which this system has no solution are collected in the following set

$$O_k = \Big\{ ((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid \neg\Big((\exists\, \mathbf{x} \in \mathbb{R}^p) \\ (\mathbf{B}\,\mathbf{x} \leq \mathbf{b}) \wedge ((\mathbf{w}_{k+1} - \mathbf{w}_k)^\top \mathbf{x} + (\theta_k - \theta_{k+1}) > 0) \Big) \Big\} \,. \quad (12.22)$$

However, O_k only takes one inequality into account, whereas (12.21) requires $q-2$ inequalities to hold simultaneously. Including these inequalities leads to the following set

$$O = \Big\{ ((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid \neg\Big((\exists\, \mathbf{x} \in \mathbb{R}^p) \\ (\mathbf{B}\,\mathbf{x} \leq \mathbf{b}) \wedge \big(\bigvee_{k=1}^{q-2} ((\mathbf{w}_{k+1} - \mathbf{w}_k)^\top \mathbf{x} + (\theta_k - \theta_{k+1}) > 0) \big) \Big) \Big\} \,. \quad (12.23)$$

Using the set O, we can now obtain a hypothesis space $\bar{G}$ that satisfies the three conditions listed at the beginning of Section 12.6.2:

$$\bar{G} = \Big\{ \psi((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid ((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \in O \Big\} \,. \quad (12.24)$$

12.6.3. Learning from compositional data with $\bar{G}$

Using a dataset $(\mathbf{X}, \mathbf{Y})$, a loss function (on the cumulative compositions) ℓ and a complexity criterion $c : \bar{G} \to \mathbb{R}^+$, the following regularized empirical loss function can be constructed

$$\begin{aligned} r_{\text{E}} : \bar{G} &\to \mathbb{R} \\ \bar{\mathbf{f}} &\mapsto \textstyle\sum_{i=1}^n \ell(\bar{\mathbf{f}}(\mathbf{x}_i), \bar{\mathbf{y}}_i) + c(\bar{\mathbf{f}}) \,. \end{aligned} \quad (12.25)$$

The minimizer of r_{E} can be found by solving the following optimization problem:

$$\underset{\bar{\mathbf{f}} \in \bar{G}}{\text{minimize}} \quad \sum_{i=1}^n \ell(\bar{\mathbf{f}}(\mathbf{x}_i), \bar{\mathbf{y}}_i) + c(\bar{\mathbf{f}}) \,.$$

Moreover, as the mapping ψ is a bijection between O and $\bar{G}$, this optimization problem can be written in a form that can be solved using numerical optimization

procedures:

$$\mathcal{M}_1: \quad \underset{(\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}}{\text{minimize}} \quad \sum_{i=1}^{n} \ell(\psi((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})(\mathbf{x}_i),\bar{\mathbf{y}}_i) + c((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})$$
$$\text{subject to} \quad ((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}) \in O\,.$$

It will be shown later in this chapter that we can easily construct a regularized empirical loss function (similar to the one presented in Eq. (12.12)) that is a convex function of $((\mathbf{w}_k)_{k=1}^{q-1},\,\boldsymbol{\theta})$. However, the implementation of the constraint $((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}) \in O$ is more complicated. Therefore, an alternative characterization of O is presented in the following subsection.

An alternative characterization of O

In this section, we present an alternative characterization of O that is amenable to be used in optimization procedures. As a starting point, we present an alternative characterization of the set O_k in the following proposition. It should be noted that the mathematical characterization that is used here is inspired on [175] who used a similar characterization to incorporate knowledge in support vector machine classifiers.

Proposition 12.3. *Let $\{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \leq \mathbf{b}\}$, with $\mathbf{B} \in \mathbb{R}^{m\times p}$ and $\mathbf{b} \in \mathbb{R}^p$, be a nonempty set. Let O_k be defined as in Eq. (12.22). We have that*

$$O_k = \Big\{((\mathbf{w}_l)_{l=1}^{q-1},\boldsymbol{\theta}) \mid (\exists \mathbf{u} \in \mathbb{R}^m)\Big(\mathbf{B}^\top\mathbf{u} - \mathbf{w}_{k+1} + \mathbf{w}_k = \mathbf{0}_p) \\ \wedge (\mathbf{b}^\top\,\mathbf{u} + \theta_k - \theta_{k+1} \leq 0) \,\wedge\, (\mathbf{u} \geq \mathbf{0}_m)\Big)\Big\}\,.$$

The proof of this proposition is rather technical and is given in Appendix 12.A (at the end of this chapter).

The proposition above can easily be generalized to obtain an alternative characterization of O.

Proposition 12.4. *Let $\{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \leq \mathbf{b}\}$, with $\mathbf{B} \in \mathbb{R}^{m\times p}$ and $\mathbf{b} \in \mathbb{R}^p$, be a nonempty set. Let O be defined as in Eq. (12.23). We have that*

$$O = \Bigg\{((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}) \mid \Big(\exists\,(\mathbf{u}_k)_{k=1}^{q-2} \in \mathbb{R}^{m\times(q-2)}\Big)\Big(\bigwedge_{k=1}^{q-2}(\mathbf{u}_k \geq \mathbf{0}_m)\Big) \\ \wedge \Big(\bigwedge_{k=1}^{q-2}(\mathbf{B}^\top\mathbf{u}_k - \mathbf{w}_{k+1} + \mathbf{w}_k = \mathbf{0}_p)\Big) \,\wedge\, \Big(\bigwedge_{k=1}^{q-2}(\mathbf{b}^\top\,\mathbf{u}_k + \theta_k - \theta_{k+1} \leq 0)\Big)\Bigg\}\,.$$

Corollary 12.5. *The set O is a convex set.*

Proof. As O can be written as the solution set of a system of affine equalities and inequalities, it is convex. □

The proposition above allows to reformulate optimization problem $\mathcal{M}_1$ as follows:

$$\begin{array}{lll} \underset{(\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta},(\mathbf{u}_k)_{k=1}^{q-2}}{\text{minimize}} & \displaystyle\sum_{i=1}^{n} \ell(\psi((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})(\mathbf{x}_i),\bar{\mathbf{y}}_i) + c((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}) & \\ \text{subject to} & \mathbf{0}_p = \mathbf{B}^\top \mathbf{u}_k - \mathbf{w}_{k+1} + \mathbf{w}_k\,, & \text{for } k=1,\ldots,q-2\,, \\ & 0 \geq \mathbf{b}^\top \mathbf{u}_k + \theta_k - \theta_{k+1}\,, & \text{for } k=1,\ldots,q-2\,, \\ & \mathbf{0}_m \leq \mathbf{u}_k\,, & \text{for } k=1,\ldots,q-2\,. \end{array}$$

A convex loss function for $\mathcal{M}_1$

As a starting point, consider the loss function that is used in the proportional odds model for compositional data with ordered components:

$$\ell(\bar{\mathbf{f}}(\mathbf{x}),\bar{\mathbf{y}}) = -\log(\bar{f}_1(\mathbf{x})^{y_1}) - \sum_{k=2}^{q} \log\left((\bar{f}_k(\mathbf{x}) - \bar{f}_{k-1}(\mathbf{x}))^{y_k}\right)\,.$$

We now have the following proposition.

Proposition 12.6. *The loss function*

$$\ell(\psi((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})(\mathbf{x}),\bar{\mathbf{y}}) = -\log(\bar{f}_1(\mathbf{x})^{y_1}) - \sum_{k=2}^{q} \log\left((\bar{f}_k(\mathbf{x}) - \bar{f}_{k-1}(\mathbf{x}))^{y_k}\right)\,,$$

where $\bar{\mathbf{f}}(.) = \psi((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})(.)$*, is a convex function of* $((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}) \in O$*.*

The proof is given in Appendix 12.A.

A complexity criterion for $\mathcal{M}_1$

In this section we derive a flexible complexity criterion that allows a smooth transition between settings with ordered components and settings in which the components are unordered. More precisely, we derive a complexity criterion c of the form

$$c((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}) = \lambda_1\, c_1((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta}) + \lambda_2\, c_2((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})\,.$$

The complexity of a function $\psi((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})$ can be measured by the squared L_2-norm of the vectors $\mathbf{w}_k$. Therefore, the first term of the complexity criterion

is

$$c_1((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) = \sum_{k=1}^{q-1} \|\mathbf{w}_k\|_2^2 \,.$$

To motivate the second term of the complexity criterion, let us compare traditional multi-class classification strategies with ordinal regression strategies. Ordinal regression can be interpreted as a restricted form of multi-class classification. Whereas a traditional multi-class classification model contains $\mathcal{O}(p\,q)$ parameters that need to be estimated, ordinal regression models only have $\mathcal{O}(p+q)$ parameters that need to be estimated. From statistical learning theory, we know that this reduction of the size of the parameter set will reduce the variance of the model (see for instance [147]). On the other hand, because of the assumptions they make, ordinal regression models have a stronger bias than multi-class classification models. In terms of predictive power, ordinal regression models will generally outperform multi-class classification models when these assumptions are (approximately) correct, as the increased bias will be dominated by the reduction of the variance. However, when these assumptions do not hold, the increased bias will generally dominate the reduction in variance and lead to poor predictive power.

The extended hypothesis space $\bar{G}$ leads to an increase in the number of parameters as compared to the proportional odds model for compositional data with ordered components. Indeed, the number of parameters is $\mathcal{O}(p\,q)$ here. However, we can use regularization to bias our model towards the proportional odds model for compositional data with ordered components. This can be achieved by penalizing the complexity of $\psi((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta})$ as follows:

$$c_2((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) = \sum_{k=1}^{q-2} (\mathbf{w}_k - \mathbf{w}_{k+1})^\top (\mathbf{w}_k - \mathbf{w}_{k+1}) \,.$$

Note that c_2 is a convex function of $((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta})$.

A reformulation of $\mathcal{M}_1$

Combining the ingredients of the previous subsections, the following optimization problem is obtained:

$$\begin{aligned}
\underset{(\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta},(\mathbf{u}_k)_{k=1}^{q-2}}{\text{minimize}} \quad & \sum_{i=1}^{n}\left(-\log(\bar{f}_1(\mathbf{x}_i)^{y_{i,1}}) - \sum_{k=2}^{q}\log\left((\bar{f}_k(\mathbf{x}_i)-\bar{f}_{k-1}(\mathbf{x}_i))^{y_{i,k}}\right)\right) \\
& + \sum_{k=1}^{q-1}\|\mathbf{w}_k\|_2^2 + \sum_{k=1}^{q-2}(\mathbf{w}_k-\mathbf{w}_{k+1})^\top(\mathbf{w}_k-\mathbf{w}_{k+1}) \\
\text{subject to} \quad & \mathbf{0}_p = \mathbf{B}^\top\mathbf{u}_k - \mathbf{w}_{k+1} + \mathbf{w}_k\,, && \text{for } k=1,\ldots,q-2\,, \\
& 0 \geq \mathbf{b}^\top\mathbf{u}_k + \theta_k - \theta_{k+1}\,, && \text{for } k=1,\ldots,q-2\,, \\
& \mathbf{0}_m \leq \mathbf{u}_k\,, && \text{for } k=1,\ldots,q-2\,,
\end{aligned}$$

where $\bar{\mathbf{f}}(.) = \psi((\mathbf{w}_k)_{k=1}^{q-1},\boldsymbol{\theta})(.)$.

12.7. Beyond convex polytopes

The power of the extended hypothesis space $\hat{G}$ heavily depends upon the appropriateness of the set B. A good set B (as defined in Eq. (12.20)) should have the following properties:

1. $X \subseteq B$.
2. $B \setminus X$ should be small.
3. B should lead to an optimization problem that can be solved efficiently.

The first property is needed to ensure that condition 2 (Section 12.6.2) holds for the set $\hat{G}$. We motivate the second property as follows. From Proposition 12.4, it can easily be seen that the size of the set O (and the resulting hypothesis space $\bar{G}$) increases when the size of B decreases. In the most extreme case, when $B = \mathbb{R}^p$, we have that $\mathbf{w}_1 = \mathbf{w}_2 = \ldots = \mathbf{w}_{q-1}$ and our extended model reduces to the proportional odds model for compositional data with ordered components. The third (logical) property can be considered to be the most limiting one. Indeed this property prevents us from choosing B equal to X. Up to now, we have been limited to let B be a convex polytope. We will use generalized inequalities to relax this limitation hereafter. Interestingly, the approach presented here can be used to generalize the setting of Fung *et al.* [175].

Generalized inequalities can be used to define (convex) sets. As an example, consider a proper cone K, an $m \times p$ matrix $\mathbf{B}$ and an m-vector $\mathbf{b}$. It can easily be shown that

$$B = \{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \preccurlyeq_K \mathbf{b}\}$$

is a convex set. For the remainder of this section, we will assume that B is defined in that way. Moreover, we assume that $X \subseteq B$.

12.7.1. A hypothesis space using proper cones

Most of the results obtained in Sections 12.6.2 and 12.6.3 can be generalized by replacing $B = \{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \leq \mathbf{b}\}$ with $B = \{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \preccurlyeq_K \mathbf{b}\}$, where K is a given proper cone. For example, Eqs. (12.23) and (12.24) can be generalized as follows:

$$O = \Big\{((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid \neg\Big((\exists\, \mathbf{x} \in \mathbb{R}^p) \\ \Big((\mathbf{B}\,\mathbf{x} \preccurlyeq_K \mathbf{b}) \wedge \big(\bigvee_{k=1}^{q-2} (\mathbf{w}_{k+1} - \mathbf{w}_k)^\top \mathbf{x} + (\theta_k - \theta_{k+1}) > 0\big)\Big)\Big)\Big\}\,, \tag{12.26}$$

and

$$\bar{G} = \Big\{\psi((\mathbf{w})_{i=1}^{q-1}, \boldsymbol{\theta}) \mid ((\mathbf{w})_{i=1}^{q}, \boldsymbol{\theta}) \in O\Big\}\,. \tag{12.27}$$

Proposition 12.7. *Given a proper cone $K \subset \mathbb{R}^p$ and its dual K^*, a matrix $\mathbf{B} \in \mathbb{R}^{m\times p}$ and an m-vector $\mathbf{b}$ such that the solution set of $\mathbf{B}\mathbf{x} \prec_K \mathbf{b}$ is not empty. Let O be defined as in Eq. (12.26). We have that*

$$O = \Big\{((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid (\exists\, (\mathbf{u}_k)_{k=1}^{q-2} \in \mathbb{R}^{m\times(q-2)})\Big(\Big(\bigwedge_{k=1}^{q-2} \mathbf{u}_k \succcurlyeq_{K^*} \mathbf{0}_m\Big) \\ \wedge \Big(\bigwedge_{k=1}^{q-2} \mathbf{B}^\top \mathbf{u}_k - \mathbf{w}_{k+1} + \mathbf{w}_k = \mathbf{0}_p\Big) \wedge \Big(\bigwedge_{k=1}^{q-2} \mathbf{b}^\top\, \mathbf{u}_k + \theta_k - \theta_{k+1} \leq 0\Big)\Big)\Big\}\,.$$

This proposition can be used to construct a set of inequality and equality constraints for optimization problem $\mathcal{M}_1$. This generalizes the original setting:

$$\begin{array}{llll} \mathcal{M}_2: & \underset{(\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}, (\mathbf{u}_k)_{k=1}^{q-2}}{\text{minimize}} & \displaystyle\sum_{i=1}^{n} \ell(\psi((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta})(\mathbf{x}_i), \bar{\mathbf{y}}_i) + c((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) & \\ & \text{subject to} & \mathbf{0}_p \;= \mathbf{B}^\top \mathbf{u}_k - \mathbf{w}_{k+1} + \mathbf{w}_k\,, & \text{for } k = 1, \ldots, q-2\,, \\ & & 0 \;\geq \mathbf{b}^\top\, \mathbf{u}_k + \theta_k - \theta_{k+1}\,, & \text{for } k = 1, \ldots, q-2\,, \\ & & \mathbf{0}_m \preccurlyeq_K^* \mathbf{u}_k\,, & \text{for } k = 1, \ldots, q-2\,. \end{array}$$

This optimization problem is a convex conic program. During the past decade, several conic programming solvers have been developed that are capable of solving a limited number of convex conic optimization problems efficiently [32, 33]. The

most important factor that determines whether a conic program can be solved with an existing convex programming solver, is the type of proper cone that is used. Most conic solvers can handle the L_2 Lorentz cone and the semidefinite cone. This means that, even though the theoretical results obtained before do not put any restrictions on the proper cones that are used, we are limited to using the L_2 Lorentz cone or the semidefinite cone.

12.7.2. An example

As an example, let us consider the case where B is a hypersphere:

$$B = \{\mathbf{x} \in \mathbb{R}^p \mid \|\mathbf{x}\|_2 \leq 1\},$$

We will now illustrate that X can be described by means of a generalized inequality. First we define the L_2 Lorentz cone K_{L_2} is given by:

$$K_{\mathrm{L}_2} = \{(\mathbf{u}, v) \in \mathbb{R}^{p+1} \mid \|\mathbf{u}\|_2 \leq v\}.$$

It can easily be seen that this cone can be used to describe B as follows:

$$B = \left\{ (\mathbf{x}, t) \in \mathbb{R}^p \times \mathbb{R} \;\middle|\; \mathrm{diag}(-\mathbf{1}_p^\top, 0) \begin{pmatrix} \mathbf{x} \\ t \end{pmatrix} \preccurlyeq_{K_{\mathrm{L}_2}} \begin{pmatrix} \mathbf{0}_p \\ 1 \end{pmatrix} \right\},$$

where $\mathbf{0}_p$ (resp. $\mathbf{1}_p$) is a p-vector of zeros (resp. ones).

It can be noted that there might exist more elegant manners to encode a hypersphere by means of a generalized inequality. However, K_{L_2} has the appealing property that it is self-dual, *i.e.* $K_{\mathrm{L}_2} = K^*_{\mathrm{L}_2}$. Therefore, the dual cone that is used in the last inequality of $\mathcal{M}_2$ is the Lorentz cone itself. This means that the resulting optimization problem can be solved using existing second order cone programming solvers.

12.8. Conclusions and discussion

In this chapter, we have introduced the problem of predicting compositional data with ordered components. The natural order on the components was translated into a partial order relation on compositions. It was argued that this order relation should be taken into account when learning to predict compositions with ordered components. An extension of the proportional odds model was used as a basis for creating those predictive models. From the results obtained from an extensive set of experiments with artificially generated data, it can be concluded that the proposed approach can outperform competing approaches that do not take the

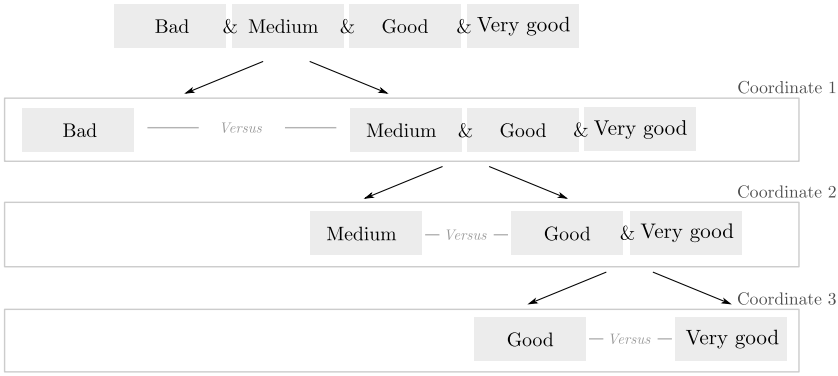


Figure 12.6: Hierarchical representation of the ordered components of a 3-part composition.

ordinal nature of the data into account.

Unfortunately, we were unable to verify if the proportional odds model for compositional data with ordered components can outperform its competitors in a real-life setting. To make such a comparison, the relationship between the inputs and the outputs in the fusarium dataset clearly was too weak to be modeled.

In Section 12.6, the proportional odds model for predicting compositions with ordered components was extended. The extended version (which can be seen as a relaxation of the original model) allows to model situations in which the latent variable motivation is slightly violated. However, the applicability of this relaxed version still needs to be supported by empirical evidence. Such evidence will require experiments on artificially generated data as well as experiments on real-life data.

The discussions in Chapters 10 and 11 may raise the question whether the principles that have been used here to encode the ordinal nature of the data can be translated to a setting that uses the Aitchison geometry. We conclude this discussion with several proposals that could serve as a link between this chapter and the previous chapter. As a starting point, consider a setting where the outputs are 3-part compositions with ordered components. Assume that these components are labeled *Bad*, *Medium*, *Good* and *Very good*. Now consider the hierarchy presented in Figure 12.6. This hierarchy suggests that three models can be built. A first model can be used to distinguish the relative amount of *Bad* from the relative amount of *Medium and Good and Very good*. A second model can be used to distinguish the relative amount of *Medium* from the relative amount of *Good and Very good*. A third model can be used to distinguish the relative amount of *Good* from the relative amount of *Very good*. If there exists some latent variable, it can be assumed that the parameter vectors these three models should be identical. Interestingly, when using the ilr-transform with a properly chosen set of basis vectors, this approach

can easily be combined with the Aitchison geometry. In this example the following basis can be used.

	$\mathbf{\Phi}_{.,1}$	$\mathbf{\Phi}_{.,2}$	$\mathbf{\Phi}_{.,3}$
Bad	$-3/\sqrt{12}$	0	0
Medium	$1/\sqrt{12}$	$2/\sqrt{6}$	0
Good	$1/\sqrt{12}$	$-1/\sqrt{6}$	$-\sqrt{2/2}$
Very good	$1/\sqrt{12}$	$-1/\sqrt{6}$	$\sqrt{2/2}$

The first coordinate in the transformed space expresses the balance between the proportional amount of *Bad* versus the proportional amount of *Medium and good*. The second coordinate expresses the proportional amount of *Medium* versus the proportional amount of *good*. Subsequently, three models with identical parameter vectors can be learned simultaneously in the ilr-transformed space. Moreover, this approach allows a relaxed form that is highly similar to the one presented in Section 12.6.

The suggestions that are made above can be considered as guidelines for merging Chapters 11 and 12. However, theoretical as well as empirical work is needed to confirm that this approach can be useful in practice.

12.A. Proofs of propositions 12.3–12.7

In this appendix, we provide the proofs of Propositions 12.3–12.7.

12.A.1. Preliminaries

As a starting point of this appendix, we summarize several results that can be found in literature and that will be used in the proofs in the following section.

Proposition 12.8 (Nonhomogeneous Farkas theorem [176])**.** *For a given $m \times n$ matrix $\mathbf{A}$, two m-vectors $\mathbf{b}$ and $\mathbf{c}$ and a scalar d, exactly one of the following two statements is true:*

- ***Statement 1:*** *The system (system 1)*

$$\begin{aligned} \mathbf{A}\,\mathbf{x} &\leq \mathbf{b} \\ \mathbf{c}^\top \mathbf{x} &> d \end{aligned}$$

  has a solotion w.r.t. $\mathbf{x}$.

- ***Statement 2:*** *The system (system 2)*

$$\mathbf{A}^\top \mathbf{y} = \mathbf{c}\,, \tag{12.28}$$
$$\mathbf{b}^\top \mathbf{y} \leq d\,, \tag{12.29}$$
$$\mathbf{y} \geq \mathbf{0}_m\,, \tag{12.30}$$

  has a solution w.r.t. $\mathbf{y}$*, or the system (system 3)*

$$\mathbf{A}^\top \mathbf{y} = \mathbf{0}_n\,, \tag{12.31}$$
$$\mathbf{b}^\top \mathbf{y} < 0\,, \tag{12.32}$$
$$\mathbf{y} \geq \mathbf{0}_m\,, \tag{12.33}$$

  has a solution w.r.t. $\mathbf{y}$.

The following corollary is a trivial consequence of the non-homogeneous Farkas theorem (it can as well be seen as an alternative way of formulating the theorem).

Corollary 12.9. *For a given $m \times n$ matrix $\mathbf{A}$, two m-vectors $\mathbf{b}$ and $\mathbf{c}$ and a scalar d, such that the solution set of $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ is not empty, the following equivalence holds:*

$$(\forall \mathbf{x} \in \mathbb{R}^n)\,(\mathbf{A}\mathbf{x} \leq \mathbf{b} \Rightarrow \mathbf{c}^\top \mathbf{x} \leq d) \iff \Big((\exists\, \mathbf{y} \in \mathbb{R}^m)(\mathbf{A}^\top \mathbf{y} = \mathbf{c} \wedge \mathbf{b}^\top \mathbf{y} \leq d \wedge \mathbf{y} \geq \mathbf{0}_m)\Big)\,.$$

The non-homogeneous Farkas theorem (NHFT) can be generalized to systems involving generalized inequalities. However, the result that can be obtained in that case is slightly weaker

Proposition 12.10 (NHFT for generalized inequalities [177])**.** *For a given $m \times n$ matrix $\mathbf{A}$, two m-vectors $\mathbf{b}$ and $\mathbf{c}$, a scalar d and a proper cone K, such that the solution set of $\mathbf{A}\mathbf{x} \preccurlyeq_K \mathbf{b}$ is not empty, the following implication holds:*

$$\Big((\exists\, \mathbf{y} \in \mathbb{R}^m)(\mathbf{A}^\top \mathbf{y} = \mathbf{c} \wedge \mathbf{b}^\top \mathbf{y} \leq d \wedge \mathbf{y} \succcurlyeq_{K^*} \mathbf{0}_m)\Big) \\ \Rightarrow (\forall \mathbf{x} \in \mathbb{R}^n)\,(\mathbf{A}\mathbf{x} \preccurlyeq_K \mathbf{b} \Rightarrow \mathbf{c}^\top \mathbf{x} \leq d)\,.$$

If additionally, we have that the solution set of $\mathbf{A}\mathbf{x} \prec_K \mathbf{b}$ is nonempty, we have that

$$(\forall \mathbf{x} \in \mathbb{R}^n)\,(\mathbf{A}\mathbf{x} \preccurlyeq_K \mathbf{b} \Rightarrow \mathbf{c}^\top \mathbf{x} \leq d) \\ \Rightarrow \Big((\exists\, \mathbf{y} \in \mathbb{R}^m)(\mathbf{A}^\top \mathbf{y} = \mathbf{c} \wedge \mathbf{b}^\top \mathbf{y} \leq d \wedge \mathbf{y} \succcurlyeq_{K^*} \mathbf{0}_m)\Big)\,.$$

12.A.2. Proofs of Propositions 12.3–12.7

Proposition 12.3 *Let $\{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \leq \mathbf{b}\}$, with $\mathbf{B} \in \mathbb{R}^{m \times p}$ and $\mathbf{b} \in \mathbb{R}^p$, be a nonempty set. Let O_k be defined as in Eq. (12.22). We have that*

$$O_k = \Big\{((\mathbf{w}_l)_{l=1}^{q-1}, \boldsymbol{\theta}) \mid (\exists \mathbf{u} \in \mathbb{R}^m)\Big(\mathbf{B}^\top \mathbf{u} - \mathbf{w}_{k+1} + \mathbf{w}_k = \mathbf{0}_p) \\ \wedge (\mathbf{b}^\top \mathbf{u} + \theta_k - \theta_{k+1} \leq 0) \wedge (\mathbf{u} \geq \mathbf{0}_m)\Big)\Big\}\,.$$

Proof. Choosing $((\mathbf{w}_l)_{l=1}^{q-1}, \boldsymbol{\theta}) \in O_k$ is equivalent to choosing $(\mathbf{w}_k, \theta_k, \mathbf{w}_{k+1}, \theta_{k+1})$ such that the following system

$$\begin{aligned} \mathbf{B}\,\mathbf{x} &\leq \mathbf{b} \\ (-\mathbf{w}_k + \mathbf{w}_{k+1})^\top \mathbf{x} + (\theta_k - \theta_{k+1}) &> 0 \end{aligned}$$

has no solution (w.r.t. $\mathbf{x}$).

As a consequence of the non-homogeneous Farkas theorem, we have that exactly one of the following statements is true:

- **Statement 1:** The system (system 1)

$$\begin{aligned} \mathbf{B}\,\mathbf{x} &\leq \mathbf{b} \\ (-\mathbf{w}_k + \mathbf{w}_{k+1})^\top \mathbf{x} + (\theta_k - \theta_{k+1}) &> 0 \end{aligned}$$

  has a solotion w.r.t. $\mathbf{x}$.

- **Statement 2:** The system (system 2)

$$\mathbf{B}^\top \mathbf{u} - \mathbf{w}_{k+1} + \mathbf{w}_k = \mathbf{0}_p\,, \tag{12.34}$$
$$\mathbf{b}^\top \mathbf{u} + \theta_k - \theta_{k+1} \leq 0\,, \tag{12.35}$$
$$\mathbf{u} \geq \mathbf{0}_m\,, \tag{12.36}$$

has a solution w.r.t. $\mathbf{u}$, or the system (system 3)

$$\mathbf{B}^\top \mathbf{u} = \mathbf{0}_p\,, \tag{12.37}$$
$$\mathbf{b}^\top \mathbf{u} < 0\,, \tag{12.38}$$
$$\mathbf{u} \geq \mathbf{0}_m\,, \tag{12.39}$$

has a solution w.r.t. $\mathbf{u}$.

From this theorem, it can easily be seen that, when system 2 has a solution, that system 1 does not have a solution. Moreover, when selecting $\mathbf{x} \in \{\mathbf{x} \mid \mathbf{B}\,\mathbf{x} \leq \mathbf{b}\}$, and solving system 3 for $\mathbf{u}$, we obtain that:

$$0 \geq \mathbf{u}^\top(\mathbf{B}^\top \mathbf{x} - \mathbf{b})\,.$$

Additionally, we have that

$$\mathbf{u}^\top(\mathbf{B}\mathbf{x} - \mathbf{b}) = \mathbf{x}^\top \mathbf{B}^\top \mathbf{u} - \mathbf{b}^\top \mathbf{u} = -\mathbf{b}^\top \mathbf{u}\,.$$

From system 3, we have that $-\mathbf{b}^\top \mathbf{u} > 0$, which contradicts our starting point.

This means that when system 1 has no solution system 2 will have a solution. This means that the set of vectors of the form $((\mathbf{w}_l)_{l=1}^{q-1}, \boldsymbol{\theta})$ for which system 2 has a solution w.r.t. $\mathbf{u}$ is identical to the set O_k. □

Remark 1. The proof given above uses form of the non-homogeneous Farkas theorem as a traditional *theorem of the alternative*. However, using Corollary 12.9 the proof trivially follows. Indeed, according to this corollary, requiring that for all $\mathbf{x} \in \{\mathbf{x} \mid \mathbf{B}\mathbf{x} \leq \mathbf{b}\}$ the following inequality holds

$$-\mathbf{w}_k^\top \mathbf{x} + \theta_k \leq -\mathbf{w}_{k+1}^\top \mathbf{x} + \theta_{k+1}\,,$$

is equivalent to requiring that there exists a $\mathbf{u} \in \mathbb{R}^m$ such that

$$(\mathbf{B}^\top \mathbf{u} - \mathbf{w}_{k+1} + \mathbf{w}_k = 0) \wedge (\mathbf{b}^\top \mathbf{u} + \theta_k - \theta_{k+1} \leq 0) \wedge (\mathbf{u} \geq \mathbf{0}_m)\,,$$

is true.

The proof of **Proposition 12.4** is a trivial extension of the proof given above.

Proposition 12.6 *The loss function*

$$\ell(\psi((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta})(\mathbf{x}), \bar{\mathbf{y}}) = -\log(\bar{f}_1(\mathbf{x})^{y_1}) - \sum_{k=2}^{q} \log\left((\bar{f}_k(\mathbf{x}) - \bar{f}_{k-1}(\mathbf{x}))^{y_k}\right) ,$$

where $\bar{\mathbf{f}}(.) = \psi((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta})(.)$*, is a convex function of* $((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \in O$*.*

Proof. Recall that O is a convex set. To prove that ℓ is convex, we will use the following properties and facts of convex functions

(i) If $\beta : \mathbb{R} \to \mathbb{R}$ is a convex (resp. concave) function, then $-\beta$ is a concave (resp. convex) function.

(ii) The nonnegative weighted sum of two convex (resp. concave) functions is convex (resp. concave).

(iii) If $\beta : \mathbb{R} \to \mathbb{R}$ is a convex function, then $\beta(\mathbf{a}^\top \mathbf{x} + b)$ is a convex function of $\mathbf{x}$.

(iv) A function is convex (resp. concave) if and only if it is convex (resp. concave) over any line segment in its domain.

(v) The function $\log(\exp(x)/(1 + \exp(x)))$ is a concave function.

By combining (v) with (i) and (iii) it can easily be seen that $-\log(\bar{f}_1(\mathbf{x})^{y_1})$ is convex.

We now show that for any $k \in \{2, \ldots, q\}$, we have that

$$-\log((\bar{f}_k(\mathbf{x}) - \bar{f}_{k-1}(\mathbf{x}))^{y_k})$$

is a convex function of $((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \in O$. When restricting this function to a line segment in its domain, the resulting univariate function has the following form

$$\alpha(t) = \log(\exp(a_1 t + b_1)/(1 + \exp(a_1 t + b_1)) - \exp(a_2 t + b_2)/(1 + \exp(a_2 t + b_2))) ,$$

where a_1, a_2, b_1, b_2 are scalars that are determined by the segment to which the restriction applies. Note that the domain of α is a closed subset of $\mathbb{R}$ (as it is a subset of O).

We now have that:

$$\frac{\mathrm{d}^2\,\alpha}{\mathrm{d}\,t^2} = \frac{e^{(a_1+a_2)t+b_1+b_2}(a_1 - a_2)^2}{(e^{a_1 t+b_1} + e^{a_2 t+b_2})^2} + \frac{a_1^2 e^{a_1 t+b_1}}{(1 + e^{a_1 t+b_1})^2} + \frac{a_2^2 e^{a_2 t+b_2}}{(1 + e^{a_2 t+b_2})^2} .$$

From this equation, we can easily see that $\frac{\mathrm{d}^2\,\alpha(t_0)}{\mathrm{d}\,t^2} > 0$, for any t_0 in the domain of α, meaning that (using property (iv)) α is convex.

When combining this result with (ii), it follows that ℓ is convex. □

Proposition 12.7 *Given a proper cone $K \subset \mathbb{R}^p$ and its dual K^*, a matrix $\mathbf{B} \in \mathbb{R}^{m \times p}$ and an m-vector $\mathbf{b}$ such that the solution set of $\mathbf{B}\mathbf{x} \prec_K \mathbf{b}$ is not empty. Let O be defined as in Eq. (12.26). We have that*

$$O = \Bigg\{ ((\mathbf{w}_k)_{k=1}^{q-1}, \boldsymbol{\theta}) \mid (\exists\, (\mathbf{u}_k)_{k=1}^{q-2} \in \mathbb{R}^{m \times (q-2)}) \Big(\Big(\bigwedge_{k=1}^{q-2} \mathbf{u}_k \succcurlyeq_{K^*} \mathbf{0}_m \Big) \\ \wedge \Big(\bigwedge_{k=1}^{q-2} \mathbf{B}^\top \mathbf{u}_k - \mathbf{w}_{k+1} + \mathbf{w}_k = \mathbf{0}_p \Big) \wedge \Big(\bigwedge_{k=1}^{q-2} \mathbf{b}^\top \mathbf{u}_k + \theta_k - \theta_{k+1} \leq 0 \Big) \Big) \Bigg\} .$$

Proof. As a starting point, consider the case where $K = \mathbb{R}^p_+$. In that case, the proposition reduces to Proposition 12.4, where the traditional non-homogeneous Farkas theorem can be used. In particular, we can use the argumentation of Remark 1 to prove the proposition. Interestingly, the generalization of the non-homogeneous Farkas theorem given in Proposition 12.10 can be used to generalize this reasoning to cases where K is a generic cone. As Proposition 12.10 generalizes Corollary 12.9 to generic proper cones K, the reasoning in Remark 1 can be applied here as well. However, to obtain the equivalence that is needed, it is required that the solution set of $\mathbf{B}\mathbf{x} \prec_K \mathbf{b}$ is not empty. □

PART V

EPILOGUE

13 General conclusions and outlook

In this chapter, the main conclusions that can be drawn from the work in this dissertation are summarized. Moreover, we highlight several aspects that may be interesting for further research. The work in this dissertation can be positioned at the crossroads of compositional data analysis and mathematical optimization. As a general approach, several data-analysis questions that may arise when working with (compositional) data of mixtures were translated into formal mathematical optimization problems and solved using numerical optimization procedures.

In part II of this dissertation, we studied the problem of selecting a subset of mixtures from a large set. This problem can be seen as a data selection problem. It turns out that several approaches exist that can be used to solve this data selection problem. However, these approaches have two potential shortcomings: (1) they lack a formal objective that is optimized, and (2) when directly applied to compositional data, they neglect the compositional nature of the data. To overcome these problems, two classes of metric-based score functions were proposed that take these issues into account. Moreover, it was shown that the incorporation of the compositional nature of the data into the objective (by using the Aitchison distance) leads to the selection of completely different subsets. These findings are supported by empirical evidence.

To find an optimal subset within a reasonable amount of time, a new Ant Colony Optimization (ACO) procedure was proposed. To improve upon existing ACO-procedures, bias-mitigating strategies were proposed and embedded into the basic Ant System procedure. Theoretical and empirical evidence illustrate that negative search bias can have a strong impact on the performance of ACO-procedures. The mitigation of this bias, using the strategies that were proposed, has a strong impact on the overall performance of ACO-procedures. Moreover, it was hinted at that the applicability of the bias-mitigating measures stretches beyond subset selection problems. It can be concluded that bias mitigation is important in practice. Therefore, it is worth considering how bias mitigation can be improved and practically be used to solve other (traditional) combinatorial optimization problems. Even though several suggestions were already made in that direction, several fundamental issues still need to be resolved. As could be concluded from the experimental sections, the incorporation of bias-mitigating strategies in MMAS (which includes for instance elitism) leads to a well-preforming procedure. Therefore, it can be expected that the incorporation of alternative ideas such as for instance local search strategies or heuristic information in the search procedure can lead to improved procedures. Nevertheless, it remains to be seen how these strategies interfere with the bias-mitigating measures.

We conclude the discussion of Part II of this dissertation by looking back at research objectives II.1–II.3. We have formally translated the mixture subset selection problem into a number of mathematical optimization problems (Objective II.1). Subsequently, we have successfully developed and analyzed (both theoretically and by means of a case study) an Ant Colony algorithm that is capable of solving these mathematical optimization problems (Objectives II.2 & II.3). We end this discussion with a point of criticism. One of the main motivations for the development of a meta-heuristic was our presumption that constructing an exact algorithm (with a tractable running time) would be extremely difficult. The successful development of a B&B procedure in Part III (for solving a hard problem), raises the question if exact approaches such as B&B could be used to solve (practical) subset selection problems as well.

In Part III of this dissertation, a set-estimator was proposed that allows to estimate the proportional contribution of an imprecisely described source to a given mixture. Several variants of this estimator were proposed and mathematical optimization problems were presented that can be used to compute this estimator in practice. Moreover, it was proven that the mathematical optimization problems that were presented can be solved efficiently. It was illustrated that this estimator is complementary to several existing estimators. Importantly, the set-estimator provides an intuitive way of handling uncertainty. It does not rely on distributional assumptions to quantify the reliability of a (point) estimate. For the past few years, there has been an evolution in several research fields towards a routinely assessment of the influence of multiple sources of uncertainty on conclusions that are drawn and decisions that are made. However, a considerable number of problem settings in these fields does not allow this uncertainty to be captured by means of probability distributions. Here, our set-estimator can be seen as a valuable alternative. Therefore, stimulating the use of this set-estimator and related approaches is probably the most challenging direction for future research. On that account, the development of a user-friendly application may be worth considering. Moreover, to solve the optimization problems that were proposed, we rely on existing numerical solvers. Even though these solvers can be considered to be the state-of-the-art, they remain (to some extent) general-purpose solvers that are not optimized to solve the optimization problems that were proposed in this dissertation. Here as well, it may be worth considering to develop special-purpose solvers that are specialized at solving the optimization problems that were proposed in Part III.

We conclude the discussion of Part III of this dissertation by looking back at research objectives III.1–III.5. The unmixing problem was formally translated into a mathematical form that led to the definition of a set estimator (Objective III.1). Subsequently, it was shown that in the case where the sources are described by convex sets, the set estimator reduces to an interval estimator. The computation of this interval led to a mathematical optimization problem (Objective III.2). Moreover, we derived an optimization problem (that was shown to be equivalent

to the former one) that can be solved efficiently and thus provides a practical way of computing our interval estimator in practice. Lastly, we studied several specific settings (high-dimensional settings and settings that require sparseness) in depth (Objectives III.3 & III.4). The real-life applicability of our set estimator (Objective III.5) was tested in a case study. From this case study, we could conclude that our set estimator can be useful in practice. However, an extensive validation of our approach would require more and larger real-life datasets. Unfortunately, it turned out that it is rather hard to find data in the public domain that can be used to validate unmixing procedures. Therefore, it is our sincerest hope that our work will stimulate the collection of those data as well as the development of novel unmixing procedures.

In Part IV of this dissertation, several learning procedures were developed that can be applied to learn (from data) a predictive model (a function) that can be used to predict compositions. Several loss functions were presented that can be used within these learning procedures. Empirical evidence was presented showing that a model that is optimal with respect to a specific loss function is likely to be sub-optimal for another loss function. Interestingly, the isometric log-ratio transform can be used to convert the problem of learning a function with compositional outputs into the more familiar multivariate regression regression problem. In this transformed setting, we focused on the incorporation of prior knowledge into the learning procedure. Three types of prior knowledge were identified and a matrix-valued kernel was proposed that can be used to jointly incorporate these types of prior knowledge. A conjugate gradient procedure was proposed that can be used to minimize the empirical regularized squared loss for that kernel. Theoretical and empirical results have shown that the incorporation of prior knowledge into the learning procedure can lead to an improved predictive performance. Nevertheless, further research is needed to gain insight into the way different types of prior knowledge interact during the learning phase. Unfortunately, in a real-life case study, we were only capable of achieving marginal improvements with respect to the predictive performance. Therefore, to be able to conclude whether the procedures that were proposed can lead to real-life improvements, more empirical evidence is needed. In numerous problem settings prior knowledge is potentially available. For example, in chemometrics, information on the molecular structure of the molecules that are considered can be a rich source of prior knowledge (for example in pointing out which wavelengths can be used to discriminate between two substances). Nevertheless, popular chemometrics software packages (such as for instance Unscrambler) do not allow this information to be incorporated. The (partly automated) inclusion of this type of knowledge could be achieved using the principles that were put forward in Part IV of this dissertation.

A linear ordering on the components of a composition can be interpreted as a special type of prior knowledge. Unfortunately, the ordering on the components does not imply a natural ordering on the compositions. A partial ordering (inspired

on stochastic dominance) was defined on compositions with ordered components. A modification of the well-known proportional odds model was proposed that allows to predict compositions with ordered components accurately. Moreover, a relaxation of this model was proposed that provides a flexible way of learning models that can be used to predict compositions with ordered components. By means of an extensive set of experiments, it is shown that taking the order on the components into account results in an improved predictive performance. Interestingly, this relaxed version leads to a conic optimization problem that can be solved efficiently. Even though the philosophy behind this relaxed form seems natural, it remains to be seen whether this procedure will lead to well-performing predictive models.

We conclude the discussion of Part IV of this dissertation by looking back at research objectives IV.1–IV.3. We studied several loss functions that can be used when learning to predict compositions (Objective IV.1). By using the ilr-transformation, the problem of learning to predict compositions was transformed into a (more traditional) multivariate regression problem. To improve the predictive performance of the multivariate regression model, we developed a general methodology that can be used to incorporate prior knowledge into the learning problem (Objective IV.2). Lastly, we developed several procedures that can be used to learn predictive models for compositional outputs with ordered components (Objective IV.3).

14 Dutch summary – Nederlandstalige samenvatting

14.1. Inleiding

In verschillende onderzoeksgebieden en industrietakken komen wetenschappers en ingenieurs in aanraking met mengsels. Daarom staan mengsels centraal in allerhande onderzoeksvragen en ingenieursproblemen. Als typevoorbeeld beschouwen we een bodemstaal. Men kan een bodemstaal (enigszins vereenvoudigd) beschouwen als een mengsel van zand, leem en klei. Een bodemkundige kan dan bijvoorbeeld geïnteresseerd zijn in het verband tussen de samenstelling van dit staal en de diepte waarop het werd genomen. Om de onderzoeksvragen die hieruit voortvloeien te kunnen beantwoorden, zal men vaak data moeten verzamelen en analyseren. Deze data zullen ondermeer bestaan uit observaties van de samenstelling van mengsels. Bijzonder aan deze data is het relatieve karakter ervan. Wanneer men bijvoorbeeld de samenstelling van een bodemstaal rapporteert zal men de aandelen zand, leem en klei vaak relatief uitdrukken tegenover een geheel (bijvoorbeeld de massa zand versus de totale massa van het bodemstaal) om zo een relatief aandeel (vaak een percentage) te bekomen. Datavectoren die men op deze manier bekomt, bevatten enkel positieve getallen die men (zonder verlies van informatie) kan herschalen zodat ze sommeren tot één. Men noemt dergelijke vectoren composities (of compositionele data). Om dergelijke data te analyseren kan men zich wenden tot technieken uit het domein van de compositionele data-analyse. In dit domein werden gedurende de laatste decennia verschillende technieken ontwikkeld die men kan aanwenden om compositionele data te analyseren. Echter, de technieken die hier ontwikkeld werden, situeren zich in de eerste plaats binnen de traditionele statistische modellering en testen van hypothesen. Een aantal probleemstellingen werden tot nog toe niet diepgaand behandeld binnen dit domein. In dit proefschrift worden een aantal (minder traditionele) probleemstellingen binnen de compositionele data-analyse behandeld, meer bepaald (1) het selecteren van subsets van compositionele datasets, (2) de propagatie van onzekerheid in de beschrijving van de componenten van een mengsel naar (de inschatting van) het proportioneel aandeel van deze componenten, en (3) predictieve modellering met compositionele outputs.

Het succes van moderne data-analysetechnieken is vaak het gevolg van een vruchtbare combinatie van de volgende basiscomponenten: probleemkennis, wiskundige optimalisatietechnieken en basisinzichten in data-analyse. In dit proefschrift vallen we dan ook vaak terug op deze basiscomponenten. Probleemstellingen worden vertaald in formele wiskundige optimalisatievraagstukken. Om deze vertaling te

kunnen maken, worden resultaten uit de compositionele data-analyse aangewend alsook een aantal inzichten uit *machine learning*. Vaak blijkt dat de optimalisatieproblemen die bekomen worden, voorbeelden zijn van meer algemeen gekende optimalisatieproblemen. Het voordeel hiervan is tweeledig. Vooreerst kunnen nieuwe optimalisatietechnieken die werden ontwikkeld ook gebruikt worden buiten de context van dit proefschrift. Daarenboven kunnen we binnen dit proefschrift ook beroep doen op bestaande optimalisatiesoftware.

Besluitend kunnen we stellen dat dit proefschrift zich bevindt op het kruispunt van compositionele data-analyse en wiskundige optimalisatie. De vertaling van concrete probleemstellingen binnen de compositionele data-analyse in formele wiskundige optimalisatieproblemen, alsook het oplossen van deze problemen vormt de kern van dit proefschrift. In de secties die hierna volgen wordt dieper ingegaan op deze probleemstellingen.

14.2. Optimale selectie van mengsels

Moderne high-throughput screening technieken laten onderzoekers toe om een groot aantal stalen te analyseren in een beperkte tijdspanne. Deze technieken geven aanleiding tot grote databanken die compositionele data bevatten. Het aanleggen van dergelijke databanken is nuttig, maar is meestal slechts een onderdeel van een groter onderzoeksproject. De verdere verwerking van deze stalen (of de data die men uit deze stalen bekomt) vergt soms meer tijd en is bijgevolg veelal duurder. In dergelijke gevallen kan het aangewezen zijn om een selectie te maken van de stalen die verdere analyse zullen ondergaan. Om deze selectie te maken kan men de gegevens die bekomen werden uit de high-throughput screening gebruiken. Uiteraard wil men de informatie-inhoud die aanwezig is binnen deze selectie zo hoog mogelijk houden. De strategie die aangewend wordt in Deel II van dit proefschrift om een informatieve selectie te maken, is gebaseerd op scorefuncties. Dergelijke functies 'scoren' de informatie-inhoud van een gegeven selectie. Vervolgens kan men de selectie kiezen die de scorefunctie maximaliseert. Door een scorefunctie te kiezen, vertaalt men het selectieprobleem in een wiskundig optimalisatieprobleem. Deze vertaling (Hoofdstuk 4), alsook het ontwikkelen van een procedure die het resulterende optimalisatieprobleem kan oplossen (Hoofdstuk 5), vormen de kern van Deel II van dit proefschrift.

In Hoofdstuk 4 van dit proefschrift worden metriekgebaseerde scorefuncties gedefinieerd. Deze functies herleiden het kiezen van een scorefunctie tot het kiezen van een metriek (of afstandsmaat). Binnen de compositionele data-analyse werden reeds verschillende afstandsmaten gedefinieerd voor composities. Deze afstandsmaten worden in Hoofdstuk 5 dan ook gebruikt om zinvolle metriekgebaseerde scorefuncties te bekomen.

Het selectieprobleem dat hiervoor beschreven werd, kan gezien worden als een voorbeeld van het (meer algemene) *subset-selection problem*. Helaas zijn dergelijke optimalisatieproblemen vaak moeilijk op te lossen. Bijgevolg doet men vaak een beroep op heuristieken. In Hoofdstuk 6 van dit proefschrift wordt een mierenkolonie-algoritme ontwikkeld (een meta-heuristiek) die men kan gebruiken om een (sub-optimale) oplossing voor het selectieprobleem te vinden. In de literatuur over mierenkolonie-algoritmen werd reeds meermaals gerapporteerd dat deze algoritmen kunnen worden misleid door wat men 'negatieve zoek-bias' noemt. Daarom werd, binnen de context van *subset-selection*, de invloed van deze zoek-bias onderzocht. Bovendien werd een nieuw mierenkolonie-algoritme ontwikkeld dat resistent is tegen deze zoek-bias. Uit een theoretische en experimentele analyse blijkt dat het voorgestelde (uitgebreide) algoritme een duidelijke verbetering biedt ten opzichte van het basis mierenkolonie-algoritme.

Tenslotte worden in Hoofdstuk 7 de scorefuncties uit Hoofdstuk 5 en het mierenkolonie-algoritme uit Hoofdstuk 6 gebruikt om een subset te selecteren uit een verzameling stalen in een praktijkvoorbeeld. Door een vergelijking te maken met een aantal andere algoritmen wordt onder meer geïllustreerd dat het ontwikkelde mierenkolonie-algoritme performant is. Bovendien wordt geïllustreerd dat het gebruik van de Aitchison-afstand (een specifieke afstandsmaat voor composities) een sterke invloed heeft op de selectie die wordt gemaakt.

14.3. Ontmengen van mengsels met imprecies beschreven componenten

Beschouw, bij wijze van inleiding, de volgende probleemstelling.

Beschouw een glas water met een zoutgehalte van 21 ppt (parts per thousand). Daarenboven weet je dat het water in dit glas een mengsel is van zoet water (zoutgehalte van 2 ppt) en zeewater (zoutgehalte van 40 ppt). Er wordt je vervolgens gevraagd wat het relatieve aandeel (uitgedrukt in ppt) aan zoet water in dit glas is.

Mits inachtname van een aantal eenvoudige basisprincipes betreffende de manier waarop mengsels worden gevormd, is een relatief aandeel van 50% het (enige) correcte antwoord op deze vraag. Deze (triviale) probleemstelling wordt enigszins complexer wanneer men stelt dat de zoutgehalten van zoet water en zeewater niet exact gekend zijn, maar binnen de intervallen $[1, 3]$, resp. $[30, 45]$, liggen (zoals op aarde het geval is). Men noemt deze beschrijving van de 'bronnen' zoet water en zeewater imprecies. In dit geval kan men voor een gegeven mengsel met een zoutgehalte van 21 ppt niet ondubbelzinnig bepalen wat het aandeel zoet water is. Daarentegen kan men wel een verzameling geven van percentages die mogelijks aanleiding kunnen geven tot een mengsel met een zoutgehalte van 21 ppt. In dit

voorbeeld kan met aantonen dat deze verzameling een gesloten interval is. Bijgevolg herleidt het bepalen van deze verzameling zich tot het bepalen van het maximum van dit interval (een maximalisatieprobleem) en het minimum van dit interval (een minimalisatieprobleem).

Als een verdere uitbreiding kan men situaties beschouwen waarin de bronnen beschreven worden door deelverzamelingen van $\mathbb{R}^n$ (i.p.v. deelverzamelingen van $\mathbb{R}$ zoals in het voorbeeld hierboven). Deze uitbreidingen vormen de kern van Deel III van dit proefschrift. In Hoofdstukken 7 en 8 van dit proefschrift wordt aangetoond dat (onder bepaalde voorwaarden) de verzameling van mogelijke percentages opnieuw een interval is. Tevens worden in deze hoofdstukken wiskundige optimalisatieproblemen gedefinieerd waarvan de oplossingen de eindpunten van dit interval beschrijven. Bovendien worden equivalente conische optimalisatieproblemen gedefinieerd die op een efficiënte manier kunnen worden opgelost (Hoofdstuk 8). Daarnaast worden een aantal uitbreidingen voorgesteld die de aanwezigheid van observatieruis op een methodologisch weldoordachte manier in rekening brengen. Tenslotte wordt in Hoofdstuk 9 aan de hand van een case-study aangetoond dat de ontwikkelde methodologie ook in de praktijk bruikbaar is.

14.4. Predictieve modellering met compositionele outputs

Een mengsel kan op verschillende manieren gekarakteriseerd worden. De samenstelling van het mengsel is waarschijnlijk de meest voor de hand liggende karakterisatie. Zo kan men bijvoorbeeld een melkstaal karakteriseren door middel van zijn vetzurensamenstelling. Anderzijds kan men datzelfde staal karakteriseren door middel van zijn (gemeten) nabije infra-rood (NIR) spectrum. Beide karakterisaties hebben een (vermoedelijk vrij sterke) verwantschap. Bijgevolg kan men trachten om de samenstelling van een mengsel te 'voorspellen' op basis van een alternatieve karakterisatie (bijvoorbeeld het NIR spectrum). In deel IV van dit proefschrift worden predictieve modelleringstechnieken ontwikkeld die kunnen aangewend worden om modellen te leren uit data (die beide karakterisaties bevat).

Het leren van predictieve modellen uit data behoort tot het domein van machine learning. Deel IV van dit proefschrift situeert zich dan ook binnen machine learning. In Hoofdstuk 10 worden verschillende verliesfuncties besproken die men kan gebruiken om predictieve modellen te leren die men kan gebruiken om composities te voorspellen. Uit de experimentele sectie in dit hoofdstuk kunnen we besluiten dat de keuze van de verliesfunctie een sterke invloed heeft op de modellen die geleerd worden.

In Hoofdstuk 11 wordt dieper ingegaan op verliesfuncties die gebruik maken van de Aitchison-afstand. Door gebruik te maken van de isometrische log-ratio

transformatie kan men een predictieprobleem waarbij de output een compositie is transformeren in een equivalent predictieprobleem waarbij de output een vector is in de Euclidische vectorruimte. Bijgevolg kan men dit probleem beschouwen als een klassiek multiple-output regressieprobleem en een beroep doen op bestaande technieken. Vaak beschikt men over bijkomende probleemkennis die men kan trachten te gebruiken om tot een model te komen dat een betere predictieve performantie heeft. In het voorbeeld hierboven kan men bijvoorbeeld vermoeden dat een aantal delen van het NIR-spectrum voornamelijk gerelateerd zijn aan de concentratie van één welbepaalde component, maar van weinig belang zijn voor de overige componenten. Echter, de huidige inzichten in het incorporeren van probleemkennis in multiple-output regressieproblemen zijn beperkt. Daarom wordt in Hoofdstuk 11 van dit proefschrift een methodologie voorgesteld die toelaat om verschillende types van domeinkennis te incorporeren in een multiple-output regressieprobleem. Hiervoor wordt gebruik gemaakt van het bestaande framework van *kernel-based vector-valued functions*.

Tenslotte wordt in Hoofdstuk 12 van dit proefschrift een probleemstelling beschouwd waarin de componenten waaruit een mengsel bestaat een natuurlijke (lineaire) orde bevatten. Deze orde kan gezien worden als een specifiek geval van domeinkennis die men tracht te incorporeren in een leerprobleem. Om deze domeinkennis te incorporeren in het leerprobleem wordt de lineaire orde op de componenten vertaald in een partiële orde op composities. Deze partiële orde is geïnspireerd op het principe van stochastische dominantie. Bovendien blijkt dat een uitbreiding van het gekende *proportional odds model* toelaat om composities met geordende componenten te voorspellen. Een experimentele evaluatie illustreert dat de voorgestelde procedure leidt tot modellen met sterke predictieve eigenschappen. Echter, de flexibiliteit van dit model is beperkt. Bijgevolg kan dit model in bepaalde situaties te rigide zijn en kan men meer performante modellen bekomen door de assumpties die het *proportional odds model* impliceert te relaxeren. Een dergelijke relaxatie wordt voorgesteld in Hoofdstuk 12 van dit proefschrift. Bovendien wordt aangetoond dat de gerelaxeerde vorm aanleiding geeft tot een conisch optimalisatieprobleem dat men efficiënt kan oplossen.

Bibliography

[1] J. Aitchison, *The statistical analysis of compositional data.* Blackburn Press, 2003.

[2] Tecator, "The Tecator dataset." http://www.dm.unibo.it/ simoncin/tecator.

[3] K. Pearson, "Mathematical contributions to the theory of evolution - on a form of spurious correlation which may arise when indices are used in the measurement of organs," *Proceedings of the Royal Society of London*, vol. 60, pp. 489–497, 1897.

[4] J. Aitchison, "The statistical analysis of compositional data (with discussion)," *Journal of the Royal Statistical Society*, vol. B44, pp. 139–177, 1982.

[5] V. Pawlowsky-Glahn, J. J. Egozcue, and R. T. D. Raimon, "Lecture notes on compositional data analysis." Available at http://dugi-doc.udg.edu/handle/10256/297.

[6] J. Aitchison, "A concise guide to compositional data analysis." Available at http://www.compositionaldata.com.

[7] J. J. Egozcue, V. Pawlowsky-Glahn, G. Mateu-Figueras, and C. Barceló-Vidal, "Isometric logratio transformations for compositional data," *Mathematical Geology*, vol. 35, pp. 279–300, 2003.

[8] J. Aitchison, *Algebraic Methods in Statistics and Probability*, ch. Simplicial Inference, pp. 1–22. American Mathematical Society, 2001.

[9] V. Pawlowsky-Glahn and J. J. Egozcue, "Geometric approach to statistical analysis on the simplex," *Stochastic Environmental Research and Risk Assessment*, vol. 15, pp. 384–398, 2001.

[10] S. P. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[11] J. Nocedal and S. J. Wright, *Numerical Optimization.* Springer, 1999.

[12] R. Horst and H. Tuy, *Global Optimization: Deterministic Approaches.* Springer, 1996.

[13] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity.* Dover Publications, 1998.

[14] C. Cortez and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[15] R. T. Rockafellar, *Convex Analysis.* Princeton University Press, 1970.

[16] J. F. Bonnans, J. C. Gilbert, C. Lemarechal, and C. A. Sagastizabal, *Numerical Optimization*. Springer, 2006.

[17] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pp. n/a–n/a, 1951.

[18] E. L. Ferguson, N. Darmon, A. Briend, and I. M. Premachandra, "Food-based dietary guidelines can be developed and tested using linear programming analysis," *The Journal of Nutrition*, vol. 134, pp. 951–957, 2004.

[19] J. Xu, M. Li, D. Kim, and Y. Xu, "RAPTOR: Optimal protein threading by linear programming," *Journal of Bioinformatics and Computational Biology*, vol. 1, pp. 95–117, 2003.

[20] F. Bartonlini, G. M. Bazzani, V. Gallerani, M. Raggi, and D. Viaggi, "The impact of water and agriculture policy scenarios on irrigated farming systems in italy: An analysis based on farm level multi-attribute linear programming models," *Agricultural Systems*, vol. 93, pp. 90–114, 2007.

[21] H. M. Markowitz, "Foundations of portfolio theory," *The Journal of Finance*, vol. 46, pp. 469–477, 1991.

[22] P. Hansen, B. Jaumard, and G. Savard, "New branch-and-bound rules for linear bilevel programming," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, pp. 1194–1217, 1992.

[23] F. A. Al-Khayyal, "Generalized bilinear programming: Part I. models, applications and linear programming relaxation," *European Journal of Operational Research*, vol. 60, pp. 306–314, 1992.

[24] J. M. Bloemhof-Ruwaard and E. M. T. Hendrix, "Generalized bilinear programming: An application in farm management," *European Journal of Operational Research*, vol. 90, pp. 102–114, 1996.

[25] A. Ben-Tal and A. Nemirovski, "Robust optimization - methodology and applications," *Mathematical Programming*, vol. 92, pp. 453–480, 2002.

[26] D. G. Luenberger, "Quasi-convex programming," *SIAM Journal on Applied Mathematics*, vol. 16, pp. 1090–1095, 1968.

[27] A. R. Conn, G. I. M. Gould, and P. L. Toint, *Lancelot: A Fortran Package for Large-Scale Nonlinear Optimization (Release A)*. Springer, 2010.

[28] A. Wächter and L. T. Biegler, "On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2006.

[29] The MathWorks Inc., "MATLAB optimization toolbox." V4.0 (R2008a).

[30] ILOG, "CPLEX optimizer." http://www.ilog.com/products/cplex.

[31] Mosek Aps, "MOSEK optimization software." http://mosek.com/ products/mosek/.

[32] J. F. Sturm, "Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11, pp. 625–653, 1999.

[33] K. C. Toh, M. J. Todd, and R. H. Tütüncü, "SDPT3 - a matlab software package for semidefinite programming, version 1.3," *Optimization Methods and Software*, vol. 11, pp. 545–581, 1999.

[34] V. R. N. Pauwels, "A multistart weight-adaptive recursive parameter estimation method," *Water Resources Research*, vol. 44, pp. n/a–n/a, 2008.

[35] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proceedings of IEEE International Conference on Neural Networks IV*, pp. 1942–1948, 1995.

[36] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley, 1989.

[37] M. F. Cardoso, R. L. Salcedo, and S. F. d. Azevedo, "The simplex-simulated annealing approach to continuous non-linear optimization," *Computers & Chemical Engineering*, vol. 20, pp. 1065–1080, 1996.

[38] L. G. Mitten, "Branch-and-bound methods: General formulation and properties," *Operations Research*, vol. 18, pp. 24–34, 1970.

[39] G. Leguizamon and Z. Michalewicz, "A new version of ant system for subset problems," *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 1459–1464, 1999.

[40] R. Hinterding, "Mapping, order-independent genes and the knapsack problem," *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 13–17, 1994.

[41] K. Scheerlinck, B. De Baets, I. Stefanov, and V. Fievez, "Subset selection from multi-experiment data sets with application to milk fatty acid profiles," *Computers and Electronics in Agriculture*, vol. 73, pp. 200–212, 2010.

[42] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Computers and Operations Research*, vol. 13, pp. 533–549, 1986.

[43] A. H. D. Brown, "Core collections: a practical approach to genetic resources management," *Genome*, vol. 31, pp. 818–824, 1989.

[44] H. De Beukelaer, P. Smýkal, G. F. Davenport, and V. Fack, "Core hunter II: fast core subset selection based on multiple genetic diversity measures using mixed replica search," *BMC Bioinformatics*, vol. 13, pp. 312–332, 2012.

[45] J. Franco, J. Crossa, S. Taba, and H. Shands, "A sampling strategy for conserving genetic diversity when forming core subsets," *Crop Sciences*, vol. 45, pp. 1035–1044, 2005.

[46] O. H. Frankel, *Genetic manipulation: impact on man and society*, ch. Genetic perspectives of germplasm conservation, pp. 161–170. Cambridge University Press, 1984.

[47] T. Jansen and T. van Hintum, "Genetic distance sampling: a novel sampling method for obtaining core collections using genetic distances with an application to cultivated lettuce.," *Theoretical and Applied Genetics*, vol. 114, pp. 421–428, 2007.

[48] R. W. Kennard and L. A. Stone, "Computer aided design of experiments," *Technometrics*, vol. 11, pp. 137–148, 1969.

[49] Camo Software AS, "The unscrambler: A multivariate statistical analysis program." http://www.camo.com/, 2013.

[50] D. Claeys, T. Verstraelen, E. Pauwels, C. Stevens, M. Waroquier, and V. Van Speybroeck, "Conformational sampling of macrocyclic alkenes using a kennard stone-based algorithm," *Journal of Physical Chemistry A*, vol. 114, pp. 6879–6887, 2010.

[51] M. Shahlaei, A. Fassihi, L. Saghaie, E. Arkan, and A. Pourhossein, "A modeling study of aldehyde inhibitors of human cathepsin k using partial least squares method," *Research in Pharmaceutical Sciences*, vol. 6, pp. 71–80, 2011.

[52] M. Daszykowski, B. Walczak, and D. L. Massart, "Representative subset selection," *Analytica Chimica Acta*, vol. 468, pp. 91–103, 2002.

[53] R. D. Clark, "Optisim: an extended dissimilarity selection method for finding diverse representative subsets," *Journal of Chemical Information Modeling*, vol. 6, pp. 1181–1188, 1997.

[54] D. J. Schoen and A. H. D. Brown, "Conservation of allelic richness in wild crop relatives is aided by assessment of genetic markers," *Proceedings of the National Academy of Sciences USA*, vol. 90, pp. 10623–10627, 1993.

[55] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[56] I. Saeys, I. Inza, and P. larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, pp. 2507–2517, 2007.

[57] S. Sahni, "Approximate algorithms for the 0/1 knapsack problem," *Journal of the Association of Computing Machinery*, vol. 22, pp. 115–124, 1975.

[58] D. Pisinger, "Heuristics for the container loading problem," *European Journal of Operational Research*, vol. 141, pp. 382–392, 2002.

[59] D. E. Goldberg, "Simple genetic algorithms and the minimal deceptive problem," in *Genetic Algorithms and Simulated Annealing* (L. Davis, ed.), (San Francisco, CA), pp. 74–88, Morgan Kaufmann Publishers Inc., 1987.

[60] M. Dorigo, G. D. Caro, and L. M. Gambardella, "Ant algorithm for discrete optimization," *Artificial Life*, vol. 5, pp. 137–172, 1999.

[61] M. Dorigo and T. Stutzle, *Ant Colony Optimization.* MIT press, Cambridge Massachusetts, 2004.

[62] T. Stutzle and H. H. Hoos, "MAX-MIN ant system," *Future Generations Computer Systems*, vol. 16, pp. 889–914, 2000.

[63] Komarudin and K. Y. Wong, "Applying ant system for solving unequal area facility layout problems," *European Journal of Operational Research*, vol. 202, pp. 730–746, 2010.

[64] C. A. Silva, J. M. C. Sousa, T. A. Runkler, and J. M. G. Sá da Costa, "Distributed supply chain management using ant colony optimization," *European Journal of Operational Research*, vol. 199, pp. 349–358, 2009.

[65] C. Blum and M. Dorigo, "The hyper-cube framework for ant colony optimization," *IEEE Transactions on Systems Man and Cybernetics Part B*, vol. 34, pp. 1161–1172, 2004.

[66] K. Scheerlinck, *Metaheurstic versus Tailor-made Approaches to Optimization Problems in the Biosciences.* PhD thesis, Ghent University, 2012.

[67] R. A. Caruana, L. J. Eshelman, and J. D. Schaffer, "Representation and hidden bias II: Eliminating defining length bias in genetic search via shuffle crossover," *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pp. 51–60, 1989.

[68] K. Vekaria and C. Clack, "Biases introduced by adaptive recombination operators," *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 670–677, 1999.

[69] F. Rothlauf and D. E. Goldberg, "Redundant representations in evolutionary computation," *Evolutionary Computation*, vol. 11, pp. 381–415, 2003.

[70] G. R. Raidl and J. Gottlieb, "Empirical analysis of locality, heritability and heuristic bias in evolutionary algorithms: A case study for the multidimensional knapsack problem," *Evolutionary Computation*, vol. 13, pp. 441–475, 2004.

[71] J. Montgomery, M. Randall, and T. Hendtlass, "Solution bias in ant colony optimization: Lessons for selecting pheromone models," *Computers and Operations Research*, vol. 35, pp. 2728–2749, 2008.

[72] J. Montgomery, M. Randall, and T. Hendtlass, "Search bias in constructive metaheuristics and implications for ant colony optimization," *Lecture Notes in Computer Science*, vol. 3172, pp. 390–397, 2004.

[73] C. Blum and M. Dorigo, "Search bias in ant colony optimization: On the role of competition-balanced systems," *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 159–174, 2005.

[74] D. Merkle and M. Middendorf, "Modelling the dynamics of ant colony optimization algorithms," *Evolutionary Computation*, vol. 10, pp. 235–262, 2002.

[75] S. Khuri, T. Back, and J. Heitkotter, "The zero/one multiple knapsack problem and genetic algorithms," *Proceedings of the ACM Symposium on Applied Computation*, pp. 188–193, 1994.

[76] M. Kong, P. Tian, and Y. Kao, "A new ant colony optimization algorithm for the multidimensional knapsack problem," *Computers and Operations Research*, vol. 35, pp. 2672–2683, 2008.

[77] S. Fidanova, "Ant colony optimization for multiple knapsack problem and model bias," *Lecture Notes in Computer Science*, vol. 3401, pp. 280–287, 2004.

[78] T. Mitchell, *Machine Learning.* McGraw-Hill, 1st ed. ed., 1997.

[79] C. Blum and M. Sampels, "When model bias is stronger than selection pressure," *Lecture Notes in Computer Science*, vol. 2439, pp. 893–902, 2002.

[80] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, vol. 344, pp. 243–278, 2005.

[81] L. E. Baum and G. R. Sell, "Growth transformations for functions on manifolds," *Pacific Journal of Mathematics*, vol. 27, pp. 211–227, 1968.

[82] R. E. Burkard, "Quadratic assignment problems," *European Journal of Operational Research*, vol. 15, pp. 283–289, 1984.

[83] E. M. Loiola, N. M. d. Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido, "A survey for the quadratic assignment problem," *European Journal of Operational Research*, vol. 176, pp. 657–690, 2007.

[84] P. C. Chu and J. E. Beasley, "A genetic algorithm for the multidimensional knapsack problem," *Journal of Heuristics*, vol. 4, pp. 63–86, 1998.

[85] S. Brooks, A. Gelman, G. L. Jones, and X. Meng, *Handbook of Markov Chain Monte Carlo.* Chapman & Hall, 2011.

[86] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[87] Y. N. T. Van Haelst, A. Beeckman, A. T. M. Van Knegsel, and V. Fievez, "Elevated concentrations of oleic acid and long chain fatty acids in milk fat of multiparous subclinical ketotic cows," *Journal of Dairy Science*, vol. 91, pp. 4683–4686, 2008.

[88] B. Vlaeminck, C. Dufour, A. M. van Vuuren, A. R. J. Cabrita, R. J. Dewhurst, D. Demeyer, and V. Fievez, "Use of odd and branched chain fatty acids in rumen contents and milk as a potential microbial marker," *Journal of Dairy Science*, vol. 88, pp. 1031–1042, 2005.

[89] N. Keshava and J. F. Mustard, "Spectral unmixing," *IEEE Signal Processing Magazine*, vol. 19, pp. 44–57, 2002.

[90] E. Christopoulou, M. Lazaraki, M. Komaitis, and K. Kaselimis, "Effectiveness of determinations of fatty acids and triglycerides for the detection of adulteration of olive oils with vegetable oils," *Food Chemistry*, vol. 84, pp. 463–474, 2004.

[91] Z. Hai and J. Wang, "Detection of adulteration in camellia seed oil and sesame oil using an electronic nose," *European Journal of Lipid Science and Technology*, vol. 108, pp. 116–124, 2006.

[92] H. Lizhi, K. Toyoda, and I. Ihara, "Discrimination of olive oil adulterated with vegetable oils using dielectric spectroscopy," *Journal of Food Engineering*, vol. 96, pp. 167–171, 2010.

[93] R. M. Maggio, L. Cerretani, E. Chiavaro, T. S. Kaufman, and A. Bendini, "A novel chemometric strategy for the estimation of extra virgin olive oil adulteration with edible oils," *Food Control*, vol. 21, pp. 890–895, 2010.

[94] F. Marini, F. Balestrieri, R. Bucci, A. D. Magri, A. L. Magri, and D. Marini, "Supervised pattern recognition to authenticate italian extra virgin olive oil varieties," *Chemometrics and Intelligent Laboratory Systems*, vol. 73, pp. 85–93, 2004.

[95] M. H. van Vliet, G. M. P. van Kempen, M. J. T. Reinders, and D. de Ridder, "Computational estimation of the composition of fat/oil mixtures containing interesterifications from gas and liquid chromatography data," *Journal of the American Oil Chemists Society*, vol. 82, pp. 707–716, 2005.

[96] S. Tompkins, J. F. Mustard, C. M. Pieters, and D. W. Forsyth, "Optimization of endmembers for spectral mixture analysis," *Remote Sensing of Environment*, vol. 59, pp. 472–489, 1997.

[97] M. A. Veganzones and M. Grana, "Endmember extraction methods: A short review," *Lecture Notes in Computer Science*, vol. 5179, pp. 400–407, 2008.

[98] D. C. Heinz and C. Chang, "Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery,"

IEEE Transactions on Geoscience and Remote Sensing, vol. 39, pp. 529–545, 2001.

[99] B. Somers, G. P. Asner, L. Tits, and P. Coppin, "Endmember variability in spectral mixture analysis: A review," *Remote Sensing of Environment*, vol. 115, pp. 1603–1616, 2011.

[100] G. J. Weltje, "End-member modeling of compositional data: Numerical-statistical algorithms for solving the explicit mixing," *Mathematical Geology*, vol. 4, pp. 503–549, 1997.

[101] P. Lu, A. Nakorchevskiy, and E. M. Marcotte, "Expression deconvolution: A reinterpretation of DNA microarray data reveals dynamic changes in cell populations," *Proceedings of the National Academy of Sciences*, vol. 100, pp. 10370–10375, 2003.

[102] Y. Zhao and R. Simon, "Gene expression deconvolution in clinical samples," *Genome Medicine*, vol. 2, pp. 93–96, 2010.

[103] W. Astle, M. De Iorio, S. Richardson, D. Stephens, and T. Ebbels, "A bayesian model of nmr spectra for the deconvolution and quantification of metabolites in complex biological mixtures," *Journal of the American Statistical Association*, vol. 107, pp. 1259–1271, 2012.

[104] C. Jutten and J. Karhunen, "Advances in nonlinear blind source separation," *Proceedings of the 4th symposium on Blind Signal Separation*, pp. 245–256, 2003.

[105] M. H. van Vliet, L. F. A. Wessels, and M. J. T. Reinders, "Knowledge driven decomposition of tumor expression profiles," *BMC Bioinformatics*, vol. 10 (Supplement 1), p. S20, 2009.

[106] N. Dobigeon, S. Moussaoui, J. Tourneret, and C. Carteret, "Bayesian separation of spectral sources under non-negativity and full additivity constraints," *Signal Processing*, vol. 89, pp. 2657–2669, 2009.

[107] P. Paatero and U. Tapper, "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values," *Environmetrics*, vol. 5, pp. 111–126, 1994.

[108] S. Moussaoui, A. Mohammad-Djafari, and C. Carteret, "Separation of non-negative mixture of non-negative sources using a bayesian approach and MCMC sampling," *IEEE Transactions on Signal Processing*, vol. 54, pp. 4133–4145, 2006.

[109] T. Bajjouk, J. Populus, and B. Guillaumont, "Quantification of subpixel cover fractions using principal component analysis and a linear programming method: Application to the coastal zone of Roscoff (France)," *Remote Sensing of Environment*, vol. 64, pp. 153–165, 1998.

[110] C. A. Bateson, G. P. Asner, and C. A. Wessman, "Endmember bundles: a new approach to incorporating endmember variability into spectral mixture analysis," *Geoscience and Remote Sensing*, vol. 38, pp. 1083–1094, 2000.

[111] S. Lee and I. E. Grossmann, "Global optimization of nonlinear generalized disjunctive programming with bilinear equality constraints: applications to process networks," *Computers & Chemical Engineering*, vol. 27, pp. 1557–1575, 2003.

[112] A. Genz and F. Bretz, *Computation of Multivariate Normal and t Probabilities.* Springer-Verlag, 2009.

[113] A. Charnes and W. W. Cooper, "Programming with linear fractional functionals," *Naval Research Logistics*, vol. 9, pp. 181–186, 1962.

[114] T. Duong, *ks: Kernel smoothing*, 2013. R package version 1.8.13.

[115] K. Shrestha and B. De Meulenaer, "Computational estimation of soybean oil adulteration in nepalese mustard seed oil based on fatty acid composition," *Communications in agricultural and applied biological sciences*, vol. 76, pp. 211–214, 2011.

[116] V. Vapnik, *The Nature of Statistical Learning Theory.* Springer-Verlag, 1995.

[117] B. Schölkopf and A. J. Smola, *Learning with Kernels - Support Vector Machines, Regularization, optimization and Beyond.* MIT Press, 2002.

[118] A. J. Izenman, *Modern Multivariate Statistical Techniques.* Springer, 2008.

[119] O. Bousquet, S. Boucheron, and G. Lugosi, "Introduction to statistical learning theory," *Lecture Notes in Computer Science*, vol. 3176, pp. 169–207, 2004.

[120] E. A. Hoerl and R. Kennard, "Ridge regression: biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, pp. 55–67, 1970.

[121] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1996.

[122] A. Agresti, *Categorical Data Analysis, 2nd version.* John Wiley and Sons, 2002.

[123] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees.* Wadsworth, 1984.

[124] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[125] C. A. Micchelli and M. Pontil, "On learning vector-valued functions," *Neural Computation*, vol. 17, pp. 177–204, 2005.

[126] L. Breiman and J. H. Friedman, "Predicting multivariate responses in multiple linear regression," *Journal of the Royal Statistical Society. Series B*, vol. 59, pp. 3–54, 1997.

[127] A. J. Izenman, "Reduced-rank regression for the multivariate linear model," *Journal of Multivariate Analysis*, vol. 5, pp. 248–264, 1975.

[128] A. van der Merwe and J. V. Zidek, "Multivariate regression analysis and canonical variates," *The Canadian Journal of Statistics*, vol. 8, pp. 27–39, 1980.

[129] H. Wold, "Soft modelling by latent variables: the nonlinear iterative partial least squares (NIPALS) approach." Perspectives in Probability and Statistics, in Honor of M. S. Barlett, pp. 117-144, 1975.

[130] S. Yu, V. Tresp, and H.-P. Kriegel, "Multi-output regularized feature projection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, pp. 1600–1613, 2006.

[131] W. Cheng and E. Hüllermeier, "Combining instance-based learning and logistic regression for multilabel classification," *Machine Learning*, vol. 76, pp. 211–225, 2009.

[132] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier, "On label dependence and loss minimization in multi-label classification," *Machine Learning*, vol. 88, pp. 5–45, 2012.

[133] G. S. Kimeldorf and G. Wahba, "Some results on Tchebycheffian spline functions," *Journal of Mathematical Analysis and Applications*, vol. 33, pp. 82–95, 1971.

[134] M. A. Álvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: a review," tech. rep., MIT, 2011.

[135] L. Baldassarre, L. Rosasco, A. Barla, and A. Verri, "Multi-output learning via spectral filtering," tech. rep., MIT, 2011.

[136] M. Hein and O. Bousquet, "Kernels, associated structures and generalizations," tech. rep., Max Planck Institute for Biological Cybernetics, 2004.

[137] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *Journal of Machine Learning Research*, vol. 6, pp. 615–637, 2005.

[138] R. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *Proceedings of the international conference on machine learning*, 2002.

[139] E. Krupka and N. Tishby, "Incorporating prior knowledge on features into learning," in *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, 2007.

[140] D. Hsu, S. M. Kakade, and T. Zhang, "Random design analysis of ridge regression," *JMLR: Workshop and Conference Proceedings*, vol. 23, pp. 9.1–9.24, 2012.

[141] L. Baldassarre, L. Rosasco, A. Barla, and A. Verri, "Vector field learning via spectral filtering," *Lecture Notes in Computer Science*, vol. 6321/2010, pp. 56–71, 2010.

[142] M. R. Hestenes and E. Stiefel, "Methods of conjugate grandients for solving linear systems," *Journal of Research of the National Bureau of Standards*, vol. 49, pp. 409–436, 1952.

[143] R. R. Grummer, "Etiology of lipid-related metabolic disorders in periparturient dairy cows," *Journal of Dairy Science*, vol. 76, pp. 3882–3896, 1993.

[144] A. Bonanome and S. M. Grundy, "Effect of dietary stearic acid on plasma cholesterol and lipoprotein levels," *New England Journal of Medicine*, vol. 318, pp. 1244–1248, 1988.

[145] I. Stefanov, *The Potential of Vibrational Spectroscopy Techniques to Determine Minor Milk Fatty Acids.* PhD thesis, Ghent University, 2012.

[146] L. Zhang, W. Zhou, and L. Jiao, "Wavelet support vector machine," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 34, pp. 34–39, 2004.

[147] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning.* Springer, 2001.

[148] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[149] P. A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds.* Princeton University Press, 2008.

[150] J. H. Manton, "Optimization algorithms exploiting unitary constraints," *IEEE Transactions on Signal Processing*, vol. 50, pp. 635–650, 2002.

[151] C. E. Windels, "Economic and social impacts of fusarium head blight: Changing farms and rural communities in the northern great plains," *Phytopathology*, vol. 90, pp. 17–21, 2000.

[152] S. Isebaert, S. De Saeger, R. Devreese, R. Verhoeven, P. Maene, B. Heremans, and G. Haesaert, "Mycotoxin-producing fusarium species occurring in winter wheat in Belgium (Flanders) during 2002-2005," *Journal of Phytopathology*, vol. 157, pp. 108–116, 2009.

[153] S. Keerthi, K. Duan, S. Shevade, and A. Poo, "A fast dual algorithm for kernel logistic regression," *Machine Learning*, vol. 61, pp. 151–165, 2005.

[154] J. Zhu and T. Hastie, "Kernel logistic regression and the import vector machine," *Journal of Computational and Graphical Statistics*, vol. 14, pp. 185–205, 2005.

[155] J. Zhu and T. Hastie, "Classification of gene microarrays by penalized logistic regression," *Biostatistics*, vol. 5, pp. 427–443, 2004.

[156] J. A. Anderson and P. R. Philips, "Regression, discrimination, and measurement models for ordered categorical variables," *Appl. Statist.*, vol. 30, pp. 22–21, 1981.

[157] G. Tutz and W. Hennevogl, "Random effects in ordinal regression models," *Computational Statistics and Data Analysis*, vol. 22, pp. 537–557, 1996.

[158] E. Frank and M. Hall, "A simple approach to ordinal classification," *Proceedings of the European Conference on Machine Learning*, pp. 145–165, 2001.

[159] W. Chu and S. Keerthi, "Support vector ordinal regression," *Neural Computation*, vol. 19, pp. 792–815, 2007.

[160] W. Waegeman, B. De Baets, and L. Boullart, "Learning layered ranking functions with structured support vector machines," *Neural Networks*, vol. 21, pp. 1511–1523, 2008.

[161] N. R. Parsons, M. L. Costa, J. Achten, and N. Stallard, "Repeated measures proportional odds logistic regression analysis of ordinal score data in the statistical software package R," *Computational Statistics and Data Analysis*, vol. 53, no. 3, pp. 632–641, 2009.

[162] P. McCullagh, "Regression models for ordinal data," *Journal of the Royal Statistical Society*, vol. 4, pp. 109–142, 1980.

[163] M. Gönen and G. Heller, "Concordance probability and discriminatory power in proportional hazards regression," *Biometrika*, vol. 92, pp. 965–970, 2005.

[164] W. Waegeman, B. De Baets, and L. Boullart, "On the scalability of ordered multi-class ROC analysis," *Computational Statistics and Data Analysis*, vol. 52, pp. 3371–3388, 2008.

[165] H. Levy, *Stochastic Dominance, Investment Decision Making under Uncertainty, 2nd Edition.* Springer, 2006.

[166] S. Greco, B. Matarazzo, and R. Słowiński, "Rough sets theory for multicriteria decision analysis," *European Journal of Operational Research*, vol. 129, pp. 1–47, 2001.

[167] W. Kotlowski, K. Dembczynski, S. Greco, and R. Słowiński, "Stochastic dominance-based rough set model for ordinal classification," *Information Sciences*, vol. 178, pp. 4019–4037, 2008.

[168] S. Lievens, B. De Baets, and K. Cao-Van, "A probabilistic framework for the design of instance-based supervised ranking algorithms in an ordinal setting," *Annals of Operations Research*, vol. 163, pp. 115–142, 2008.

[169] M. Rademaker and B. De Baets, "Optimal restoration of stochastic monotonicity with respect to cumulative label frequency loss functions," *Information Sciences*, vol. 181, pp. 747–757, 2011.

[170] E. Van Broekhoven, V. Adriaenssens, and B. De Baets, "Interpretability-preserving genetic optimization of linguistic terms in fuzzy models for fuzzy ordered classification: An ecological case study," *International Journal of Approximate Reasoning*, vol. 44, pp. 65–90, 2007.

[171] A. M. Mouton, B. De Baets, E. Van Broekhoven, and P. L. M. Goethals, "Prevalence-adjusted optimisation of fuzzy models for species distribution," *Ecological Modelling*, vol. 24, pp. 982–993, 2009.

[172] R Development Core Team, *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

[173] C. G. Broyden, "The convergence of a class of double-rank minimization algorithms: 2. the new algorithm," *IMA Journal of Applied Mathematics*, vol. 6, pp. 222–231, 1970.

[174] W. Waegeman, J. Verwaeren, B. Slabbinck, and B. De Baets, "Supervised learning algorithms for multi-class classification problems with partial class memberships," *Fuzzy Sets and Systems*, vol. 184, pp. 106–125, 2011.

[175] G. M. Fung, O. L. Mangasarian, and J. W. Shavlik, "Knowledge-based support vector machine classifiers," in *Advances in Neural Processing Systems*, 2002.

[176] O. L. Mangasarian, *Nonlinear Programming.* McGraw-Hill, 1969.

[177] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications.* MPS-SIAM Series on Optimization, 2001.

Curriculum Vitae

Personalia

Name	Jan Verwaeren
Gender	Male
Date of birth	25/09/1984
Place of birth	Dendermonde, Belgium
E-mail	Jan.Verwaeren@UGent.be

Education

University

- 2006 - 2007

  Third year of second cycle Bio-engineer in Land management and forestry (option management of forest and nature), Ghent University.

- 2005 - 2006

  Second year of second cycle Bio-engineer in Land management and forestry (option management of forest and nature), Ghent University.

- 2004 - 2005

  First year of second cycle Bio-engineer in Land management and forestry, Ghent University.

- 2002 - 2004

  First cycle Bio-engineer, Ghent University.

Secondary school

- 1996 - 2002:

  Heilig Maagdcollege Dendermonde, Dendermonde, Belgium

Employment

- 2007 - present

  Teaching assistant and PhD student at research unit KERMIT, department of Mathematical modelling, statistics and bioinformatics, Ghent University.

Teaching experience

As a teaching assistant (2007-present) I have been involved in multiple courses organized by the department of Mathematical modelling, statistics and bioinformatics. These courses can all be situated in the broad field of applied mathematics (and scientific programming).

- Mathematics 3: Differential equations, 2007-2012 (Prof. B. De Baets)
- Mathematics 4: Probabilistic modeling, 2007-2014 (Prof. B. De Baets)
- Computational modeling, 2007-2014 (Prof. B. De Baets)
- Predictive modeling (Machine Learning), 2009-2014 (Prof. W. Waegeman and Prof. B. De Baets)
- Quality control and risk analysis, 2007-2014 (Prof. O. Thas)
- Modeling and simulation of biological systems, 2007-2009 (H. Spanjers and Prof. I. Nopens)

Scientific output

Publications in international journals (ISI-papers)

- Van Der Weeën, P., **Verwaeren, J.**, and De Baets, B. (2014). A search grid for parameter optimization as a byproduct of model sensitivity analysis. Submitted to: APPLIED MATHEMATICS AND COMPUTATION.

- Jorjong, S., van Knegsel A., **Verwaeren, J.**, De Baets, B., Kemp, B., Bruckmaier, R., and Fievez, V. (2014). Milk fatty acids as Possible biomarkers to early diagnose high concentration of blood plasma non-esterified fatty acids and subclinical ketosis in dairy cows. Submitted to: JOURNAL OF DAIRY SCIENCE.

- Giustarini, L., Vernieuwe, H., **Verwaeren, J.**, Chini, M., Hostache, R., Matgen, P., Verhoest, N.E.C., and De Baets, B. (2014). Accounting for image uncertainty in SAR-based flood mapping. Submitted to: INTERNATIONAL JOURNAL OF APPLIED EARTH OBSERVATION AND GEOINFORMATION.

- Fukuda, S., De Baets, B., Waegeman, W., **Verwaeren, J.**, and Mouton, A. (2013). Habitat prediction and knowledge extraction for spawning European grayling (Thymallus thymallus L.) using a broad range of species distribution models. ENVIRONMENTAL MODELLING & SOFTWARE, 47, 1-6.

- **Verwaeren, J.**, Scheerlinck, K., and De Baets, B. (2013). Countering the negative search bias of Ant Colony Optimization in subset selection problems. COMPUTERS & OPERATIONS RESEARCH, 40(4), 931-942.

- **Verwaeren, J.**, Waegeman, W., and De Baets, B. (2012). Learning partial ordinal class memberships with kernel-based proportional odds models. COMPUTATIONAL STATISTICS & DATA ANALYSIS, 56(4), 928-942.

- Van der Weeën, P., Baetens, J., **Verwaeren, J.**, Van Doorslaer, X., Heynderickx, P., Dewulf, J., and De Baets, B. (2012). Modeling the photocatalytic degradation of moxifloxacin by means of a stochastic cellular automaton. CHEMICAL ENGINEERING JOURNAL, 188, 181-190.

- Waegeman, W., **Verwaeren, J.**, Slabbinck, B., and De Baets, B. (2011). Supervised learning algorithms for multi-class classification problems with partial class memberships. FUZZY SETS AND SYSTEMS, 184(1), 106-125.

Conference proceedings

- **Verwaeren, J.**, Waegeman, W., Pahikkala, T., Airola, A., and De Baets, B. (2012). Incorporating prior knowledge in multiple output regression with kernel-based vector functions. Proceedings of the 21th machine learning conference of Belgium and The Netherlands. Presented at the 19th Conference on Machine Learning in the Benelux (Benelearn 2012), Ghent, Belgium: Ghent University.

- **Verwaeren, J.**, Waegeman, W., and De Baets, B. (2010). Ensemble methods for multi-label learning of compositional data. In G. Tsoumakas (Ed.),

Proceedings of the 26th international conference on machine learning. Presented at the 2nd International workshop on Learning from Multi-label Data (MLD 10).

- **Verwaeren, J.**, Waegeman, W., and De Baets, B. (2010). Learning partial ordinal class memberships in a proportional odds setting. Proceedings of the 19th machine learning conference of Belgium and The Netherlands. Presented at the 19th Conference on Machine Learning in the Benelux (Benelearn 2010), Louvain, Belgium: University Liége, DTAI research group.

Conference abstracts

- **Verwaeren, J.**, Rademaker., M., and De Baets, B. (2013). A Set-Based Approach to the Decomposition of Linear Mixtures Using Quasi-Convex Programming. Presented at the 27th annual conference of the Belgian Operational Research Society. Kortrijk, Belgium.
- **Verwaeren, J.**, Waegeman, W., and De Baets, B. (2011). Incorporating prior knowledge in multiple output regression with application to chemometrics. Poster presentation at the 20th Belgian Dutch conference on Machine Learning (BeNeLearn 2011). The Hague, The Netherlands.
- Waegeman, W., **Verwaeren, J.**, and De Baets, B. (2009). Learning ordinal partial class memberships with kernel-based proportional odds models. Presented at the 3rd International conference on Computational and Financial Econometrics (CFE 09). London, United Kingdom.
- **Verwaeren, J.**, and De Baets, B. (2009). How to counter ACO's negative search bias in subset selection problems? Presented at the 23rd annual conference of the Belgian Operational Research Society. Leuven, Belgium.