7

MONOGRAPH SERIES: MAREK GĄGOLEWSKI - DATA FUSION. THEORY, METHODS, AND APPLICATIONS

7

MONOGRAPH SERIES

MAREK GĄGOLEWSKI

DATA FUSION. THEORY, METHODS, AND APPLICATIONS

INSTITUTE OF COMPUTER SCIENCE
POLISH ACADEMY OF SCIENCES

MONOGRAPH SERIES
INFORMATION TECHNOLOGIES: RESEARCH
AND THEIR INTERDISCIPLINARY APPLICATIONS

7

MAREK GĄGOLEWSKI

DATA FUSION
THEORY, METHODS, AND APPLICATIONS

INSTITUTE OF COMPUTER SCIENCE
POLISH ACADEMY OF SCIENCES

Warsaw, 2015

Publication issued as a part of the project:
"Information technologies: research and their interdisciplinary applications",
Objective 4.1 of Human Capital Operational Programme.
Agreement number UDA-POKL.04.01.01-00-051/10-01.

Publication is co-financed by European Union from resources of European Social Fund.

Project leader: Institute of Computer Science, Polish Academy of Sciences

Project partners: System Research Institute, Polish Academy of Sciences, Nałęcz Institute of Biocybernetics and Biomedical Engineering, Polish Academy of Sciences

Editors-in-chief: Olgierd Hryniewicz
Jan Mielniczuk
Wojciech Penczek
Jacek Waniewski

Reviewers: Gleb Beliakov, Radko Mesiar

Marek Gągolewski
Systems Research Institute Polish Academy of Sciences
ul. Newelska 6, 01-447 Warsaw, Poland email:
gagolews@ibspan.waw.pl
http://www.ibspan.waw.pl/~gagolews

Publication is distributed free of charge

ISBN 978-83-63159-20-7

Cover design: Waldemar Słonina

Introductory exercise

Please:

1. Grab two pencils, one in each hand.
2. With one eye closed, try to touch the erasers together. Was that difficult?
3. Now try it with both eyes open. Was that easier?

Humans are quite unusual in that both their eyes face forward and that they have overlapping visual fields. This kind of sensor redundancy and the nature of the information fusion process applied by the human brain makes 3D stereo vision possible. Readers who found touching two pencils much harder with an eye shut already got an impression of the importance of data fusion and may proceed to page 27. Otherwise, in the Preface we shall explore the role of data fusion in various real-world applications.

Preface

Appropriate fusion of large, complex data sets is necessary in the information era. Having to deal with just a few records already forces the human brain to look for patterns in the data and to make its overall picture instead of conceiving a reality as a set of individual entities, which are much more difficult to process and analyze. Quite similarly, the usage of appropriate methods to reduce the information overload on a computer, may not only increase the quality of the results but also significantly decrease algorithms' run-time.

It is known that information systems relying on a single information source (e.g., measurements gathered from one sensor, opinions of just a single authoritative decision maker, outputs of one and only one machine learning algorithm, answers of an individual social survey taker) are most often neither accurate nor reliable.

The theory of aggregation is a relatively new research field, even though various particular methods for data fusion were known and used already by the ancient mathematicians. Since the 1980s, studies of aggregation functions most often focus on the construction and formal, mathematical analysis of diverse ways to summarize numerical lists with elements in some real interval $\mathbb{I} = [a, b]$. This covers different kinds of broadly-conceived means, fuzzy logic connectives (t-norms, fuzzy implications), as well as copulas. Quite recently, we observe an increasing interest in aggregation on partially ordered sets – in particular, on ordinal (linguistic) scales.

Among the seminal monographs on the applied mathematics-oriented *classical* aggregation theory there are *Aggregation Functions: A Guide for Practitioners* [49] by Beliakov, Pradera, and Calvo and *Aggregation Functions* [230] by Grabisch, Marichal, Mesiar, and Pap. We note that the typical mathematical *arsenal* used by aggregation theoreticians consists of a very creative combination of approaches known from, among others, algebra, calculus, order and measure theory (in fact, aggregation theory results strongly contribute to these subfields as well). What is more, particular subclasses of aggregation functions are studied in-depth in the following textbooks: *Triangular norms* [277] authored by

Klement, Mesiar, and Pap, *Fuzzy implications* [18] by Baczyński and Jayaram, *Handbook of means and their inequalities* [87] by Bullen, as well as – very recently – *A Practical Guide to Averaging Functions* [39] by Beliakov, Bustince, and Calvo. We shall also mention the book by Torra and Narukawa (*Modeling Decisions: Information Fusion and Aggregation Operators* [449]), which perhaps is the most computer science-oriented work of the ones listed. However, in [49] and [39] numerous interesting algorithms and computational issues are discussed too.

During the 2013 AGOP – *International Summer School on Aggregation Operators* – conference in Pamplona, Spain, Prof. Bernard De Baets in his plenary lecture [137] pointed out the need to convey research on the so-called *Aggregation 2.0*. Of course, Aggregation 2.0 does not aim to replace or in any terms depreciate the very successful and important *classical* aggregation field, but rather to attract the investigators' attention to new, more complex domains, most of which cannot be properly handled without using computational methods. From this perspective, data fusion tools may be embedded in larger, more complicated information processing systems and thus studied as their key components.

A proper complex data fusion has been of interest to many researchers in diverse fields, including computational statistics, computational geometry, bioinformatics, machine learning, pattern recognition, quality management, engineering, statistics, finance, economics, etc. Let us note that it plays a crucial role in:

— a synthetic description of data processes or whole domains,

— creation of rule bases for approximate reasoning tasks,

— consensus reaching and selection of the optimal strategy in decision support systems,

— missing values imputations,

— data deduplication and consolidation,

— record linkage across heterogeneous databases,

— automated data segmentation algorithms' construction (compare, e.g., the k-means and hierarchical clustering algorithms).

We observe that many useful machine learning methods are based on a proper aggregation of information entities. In particular, the class of ensemble methods for classification is very successful in practice because of the assumption that no single "weak" classifier can perform as well as their whole group. Interestingly, many of the winning solutions to data mining competitions on *Kaggle* and similar platforms base somehow on the random forest and similar algorithms. What is more, e.g., neural networks – universal approximators – and other deep learning tools can be understood as hierarchies of individual fusion functions. Thus, they can be conceived as kinds of aggregation techniques as well. We should also

mention that an appropriate data fusion is crucial to business enterprises. For numerous reasons, companies are rarely eager to sell large parts of the data sets they posses to their clients. Instead, only carefully pre-processed and aggregated data *models* are delivered to the customers.

This monograph is a first attempt to integrate the spread-out results from different domains using the methodology of the well-established *classical* aggregation framework, introduce researchers and practitioners to Aggregation 2.0, as well as to point out the challenges and interesting directions for further research. It is organized as follows.

— In *Chapter 1* we review classical aggregation results which deal with aggregation of numeric tuples with elements in some real interval $\mathbb{I} = [a, b]$ or $]a, b[$. We list some interesting properties of fusion functions on such a domain which may be crucial in various practical applications. Even though the described data model seems to be quite simple at a first glance, it shall provide us with a deep insight on the nature of more complex data fusion processes. In particular, we pay special attention to the notion of monotonicity.

Then we discuss general construction methods that may be used to derive new fusion functions from simpler ones. Additionally, we present the connection between aggregation functions and monotone (fuzzy) measures and integrals as well as introduce the notion of a penalty-based and an extended fusion function.

Further on we present different ways which can aid in an appropriate tool selection for diverse tasks. This includes characterization theorems, synthetic numerical characteristics, as well as algorithms to learn fusion functions from empirical data.

Moreover, the topic of aggregation of data on an ordinal scale and – more generally – bounded partially ordered sets, as well as on a nominal scale is presented.

— *Chapter 2* deals with aggregation of d-dimensional data, this time for $d > 1$. Our point of departure consists of data fusion tools which are studied in fields such as computational statistics and computational geometry. Among their important properties we find, e.g., equivariances to particular geometrical transformations, as well as generalizations of some of the properties studied in the previous chapter. We note that the simplest fusion functions may be constructed by means of componentwise extensions of one-dimensional mappings. Other ones are based on the concept of data depth or penalty minimizers.

We are also interested in aggregation on product lattices and character sequences, especially in connection with the Hamming distance.

— In *Chapter 3* we focus on the topic of strings' aggregation, that is tuples of not necessarily conforming lengths. In this case, various ordering relations

may be defined, e.g., the lexicographic order. The data types of our interest include numeric strings which represent informetric data, as well as character strings, like DNA and protein sequences. It turns out that the most influential data fusion methods on such a domain may be expressed as minimizers of various string distance-based penalties. Because of that, we include a comprehensive overview of character string metrics. This embraces the notion of a generic edit, q-gram, and Dinu rank distance.

— *Chapter 4* deals with aggregation of much more complex data types: directional data, real intervals, fuzzy numbers, random variables, graphs and relations, as well as heterogeneous data sets. We shall observe that some of the key ideas in data fusion can be extrapolated to these kinds of data models.

— Finally, in *Chapter 5*, we discuss various numerical characteristics of different objects. This topic is inevitably connected to data aggregation. In particular, we are interested in a synthetic description of probability distributions, spread of numeric lists, decision makers' consensus, economical inequity, informetric data, fuzzy numbers, and fusion functions themselves. We end the chapter with a discussion on the so-called checksums, which – as it shall turn out – require a quite different treatment than other measures.

— In the *Appendix*, following the excellent approach from [49], we provide the implementations of the most interesting algorithms. For that, we use the R [397] and C++11 programming language. In the latter case, the Rcpp package classes [177] are used as a link between these two languages.

Apart from the provision of a global and concise view on fusion functions across different domains ("Aggregation 2.0"), original contributions in this monograph, which were not yet published at the time of its writing, include, but are not limited to:

— *Chapter 1*: The idea of incremental fusion functions as a generalization of recursive aggregation tools (Definition 1.121); new methods for learning aggregation operators from empirical data, including the least Chebyshev metric fitting tasks in Section 1.6.1, the least squares error fitting with output ranking preservation in Section 1.6.2.B, applications of weights' regularization to prevent model overfit, fitting weights to quasi-arithmetic means (without variables' linearization); some notes on aggregation of elements on nominal scale in Section 1.8.

— *Chapter 2*: Extension of results published in [208] concerning aggregation of d-dimensional real tuples, including Propositions 2.13, 2.14, 2.19, 2.24, 2.32, and 2.30; a construction of SVD-based similarity transform equivariant fusion functions in Sec. 2.2.3; proposal of a framework for penalty-based multidimensional fusion functions in Sec. 2.5.5 and their general properties (in particular, Proposition 2.54); a new evolutionary algorithm for approximating the Hamming distance-based 1-center character sequence.

— *Chapter 3*: New results concerning aggregation of informetric data (Proposition 3.8 and 3.14), proposal for a list of desirable properties that such data fusion tools should fulfill, new aggregation methods for numeric strings in Section 3.2.3, including the 1-median for informetric data under assumption that $\mathbb{I} = [0, \infty]$; an exact algorithm to compute a centroid of two character strings as well as an evolutionary algorithm for 1-median of arbitrary number or character strings with respect to the Levenshtein distance, a list of desirable properties of fusion functions for character sequences and strings in Section 3.3.

— *Chapter 4*: Fast approximate set exemplar search algorithm in arbitrary finite semimetric spaces in Section 4.6.

— *Chapter 5*: A generalization of a spread relation [209] in Section 5.2.3 for multidimensional numeric lists and a list of new spread measures' construction methods.

The author would like to thank Prof. Gleb Beliakov, Prof. Radko Mesiar, and Dr. Simon James for the useful, in-depth comments on the manuscript and to Prof. Olgierd Hryniewicz who encouraged him to write this book in November 2014. Moreover, he wishes to thank Prof. Bernard De Baets and Prof. Janusz Kacprzyk for motivating him to convey research on Aggregation 2.0. Also, the help of his Ph.D. students Anna Cena and Maciej Bartoszuk while dealing with early versions of this work is much appreciated. He is also indebted to Prof. Martin Štěpnička and other researchers with the Institute for Research and Applications of Fuzzy Modeling for their great hospitality during a research visit at the University of Ostrava, Czech Republic during which he has written some key parts of this monograph.

The study was cofounded by the European Union from resources of the European Social Fund, Project PO KL "Information technologies: Research and their interdisciplinary applications", Agreement UDA-POKL.04.01.01-00-051/10-00 as well as by the research task A4.1.2/2015, "Algorithms for data aggregation and fusion", Systems Research Institute, Polish Academy of Sciences and by the National Science Center, Poland, research project 2014/13/D/HS4/01700 (research in Sections 3.2 and 5.4).

Marek Gagolewski
Warsaw, December 2015

Contents

List of Figures

List of Tables

Notation convention and R basics

In this book we roughly follow the conventions used in [230], which are to some degree consistent with the way the R [397] environment handles vector and matrix computations. In particular:

— The set of natural numbers, $\{1, 2, \dots\}$, is denoted by $\mathbb{N}$, by $\mathbb{N}_0$ we mean the set $\mathbb{N} \cup \{0\}$, and the set of all integers is denoted with $\mathbb{Z}$. Additionally, $\mathbb{R}$ is the set of reals, $\mathbb{R}_+ =]0, \infty[$, and $\mathbb{R}_{0+} = [0, \infty[$. Where it is needed, $\bar{\mathbb{R}} = [-\infty, \infty]$ denotes the set of extended reals. By default, we assume that $+\infty + (-\infty) = -\infty$ and $0 \cdot \infty = 0$ (unless stated otherwise).

— The interval closure of the set $S \subseteq \mathbb{R}$, i.e., the smallest closed interval that contains S, is denoted with $\operatorname{intcl} S$. For any $x \in \mathbb{R}$, $\lfloor x \rfloor = \max\{y \in \mathbb{Z} : y \leq x\}$ and $\lceil x \rceil = \min\{y \in \mathbb{Z} : y \geq x\}$ denote the floor and ceiling function, respectively.

— For any natural number n, let $[n] = \{1, 2, \dots, n\}$, with convention $[0] = \emptyset$. Moreover, $[i : j] = \{i, i + 1, \dots, j\}$ for any $i \leq j$. Thus, $[n] = [1 : n]$. Here is a corresponding R code:

```
seq_len(0)          # [0]
## integer(0)
seq_len(5)          # [5]
## [1] 1 2 3 4 5
0:4                 # [0:4]
## [1] 0 1 2 3 4
```

— Given a set X, let $X^* = \bigcup_{n=2}^{\infty} X^n$ denote the set of all sequences with elements in X of length at least (if not stated explicitly otherwise) two.

— Each sequence is denoted with a bold symbol, e.g., $\mathbf{x} = (x_1, \dots, x_n)$. Note that in each case we use 1-based indexing (like in the R programming language). Having that in mind is crucial when it comes to implementing algorithms to perform computations on vectors: for instance, languages like C++, Java, and Python use 0-based indexing.

```
x <- c(2, 4, 6, 8)
n <- length(x)      # n == 4
x[1]                # the first element in x
## [1] 2
x[n]                # the last element in x
## [1] 8
```

— Given arbitrary $x \in X$, by $(n * x)$ we denote an n-tuple (a sequence of length n) $(x, x, \ldots, x) \in X^n$. More generally, $(n * (x_1, \ldots, x_k)) = (x_1, \ldots, x_k, \ldots, x_1, \ldots, x_k) \in X^{nk}$ denotes the fact that $(x_1, \ldots, x_k)$ is repeated exactly n times, with recycling.

```
rep(1, 5)           # (5 * 1)
## [1] 1 1 1 1 1
rep(1:2, 3)         # (3 * (1, 2))
## [1] 1 2 1 2 1 2
```

— Given, say, $\mathbf{x} \in X^n, \mathbf{y} \in X^m, t \in X$, $(\mathbf{x}, \mathbf{y}, t) \in X^{n+m+1}$ denotes their concatenation into a single vector.

```
x <- c(1, 2, 3)
y <- c(4, 5)
t <- 6
c(x, y, t)           # (x, y, t)
## [1] 1 2 3 4 5 6
```

— Binary operations like $+, -, \cdot, /, \wedge$ (minimum), and $\vee$ (maximum) on vectors of equal lengths n are applied elementwise and thus output a vector of length n too. On the other hand, if one of the operands is a scalar, then it is extended to a vector of length n in such a way that $\mathbf{x} + t = \mathbf{x} + (n * t)$.

```
c(-1, 1, -2, 2) * c(1, 2, 3, 4)  # vector * vector
## [1] -1  2 -6  8
2 * c(1, 3, 5)                   # scalar * vector
## [1]  2  6 10
```

Note that in fact in R there are no separate scalar data types: single values are represented as vectors of length 1.

— If $\mathbf{A} \in \mathbb{R}^{d \times n}$ is a matrix with d rows and n columns and $\mathbf{t} \in \mathbb{R}^d$, then by, e.g., $\mathbf{A} + \mathbf{t}$ we mean $\mathbf{A} + [\mathbf{t}\ \mathbf{t}\ \cdots\ \mathbf{t}]$, i.e., $\mathbf{t}$ is treated as a column vector. Moreover, $\mathbf{A} + t = \mathbf{A} + (d * t) = \mathbf{A} + [(d * t)\ \cdots\ (d * t)]$.

```
d <- 2
n <- 3
A <- matrix(byrow=TRUE, nrow=d, ncol=n,
            c(1, 2, 3,
              4, 5, 6))
A
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
```

```
A * c(-1, 1)
##      [,1] [,2] [,3]
## [1,]   -1   -2   -3
## [2,]    4    5    6

A / 2
##      [,1] [,2] [,3]
## [1,]  0.5  1.0  1.5
## [2,]  2.0  2.5  3.0
```

— Regarding evaluation of n-argument functions, we interchangeably use notations: $\mathsf{F}(x_1, \ldots, x_n) = \mathsf{F}((x_1, \ldots, x_n)) = \mathsf{F}(\mathbf{x})$. If F is defined on a domain X, then for $Y \subset X$, $\mathsf{F}|_Y$ denotes the projection of F onto Y (domain restriction).

— If a function F is defined on X, then we implicitly assume that it may be extended onto X^n by vectorization: $\mathsf{F}(x_1, \ldots, x_n) = (\mathsf{F}(x_1), \ldots, \mathsf{F}(x_n))$.

```
sign(c(-2, 1, 0, 0.5))
## [1] -1  1  0  1
```

On a side note, if vectorization is not an R function's inherent feature, we can assure it manually by calling a functional programming construct called `sapply()`.

```
sapply(c(-2, 1, 0, 0.5), sign)
## [1] -1  1  0  1
```

— $\mathbf{1}(p)$ denotes the Boolean indicator function, $\mathbf{1}(p) = 1$ whenever a logical statement p is true and 0 otherwise. Moreover, the characteristic function is denoted with $\mathbf{1}_X(x) = \mathbf{1}(x \in X)$ for any set X. Of course, these functions may be vectorized if needed.

```
x <- c(-2, 1, 0, 0.5)
as.integer(x > 0)
## [1] 0 1 0 1
```

— For any finite set X, $|X|$ denotes its cardinality. If $\mathbf{x}$ is a sequence, then the same notion, $|\mathbf{x}|$, is used to denote its length.

— Let $\mathfrak{S}_Y$ denote the set of all permutations of a finite set Y. Given $\mathbf{x} \in X^n$ and $\sigma \in \mathfrak{S}_{[n]}$ let $\mathbf{x}_\sigma := (x_{\sigma(1)}, \ldots, x_{\sigma(n)})$. Additionally, let $x_{(i)}$ denote the ith *order statistic* in a vector $\mathbf{x} \in X^n$, i.e., the ith smallest value in that vector. The term "smallest" is of course relative to some linear order $\leq$ on X. For instance, if $X = \mathbb{R}$, we use (if not stated otherwise) standard ordering of reals. Thus, it holds:

$$x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(i)} \leq \cdots \leq x_{(n)}$$

Of course, $x_{(i)} = x_{\sigma(i)}$, where σ is a so-called *ordering permutation* of $\mathbf{x}$. Generally (if there are tied observations in $\mathbf{x}$) such a permutation might

be ambiguous, so we assume that σ is the *stable ordering permutation*: for $\mathbf{x} = (1, 2, 1, 2, 1)$ we always get $\sigma = (1, 3, 5, 2, 4)$. The linear order $\sqsubseteq$ used here is such that $x_i \sqsubseteq x_j$ whenever $x_i < x_j$ or $(x_i = x_j$ and $i \leq j)$.

```
x <- c(13, 11, 12, 11, 11)
o <- order(x) # a (stable) ordering permutation
o
## [1] 2 4 5 3 1
x[o[1]] # the smallest value in x
## [1] 11
x[o[5]] # the largest value in x
## [1] 13
```

— The uniform distribution on a set A is denoted with $\mathrm{U}A$, e.g., $\mathrm{U}[0, 1]$ or $\mathrm{U}\{-1, 1\}$. The normal distribution with expected value of μ and standard deviation of σ is denoted with $\mathrm{N}(\mu, \sigma)$.

Regardless of the differences in vector indexing in the Python programming language, similar code chunks could have been provided for `ndarray`s defined in the NumPy package.

Let us also note that how C++ code can seamlessly be integrated in R (for instance, to speed up computations, access external libraries, or make use of lower-level programming concepts, like dynamic data structures). For that, we use the Rcpp package [177].

C++ source files may be turned into a dynamically linked library (automatically loaded by R) via a call to:

```
Rcpp::sourceCpp('filename.cpp')
```

For quite simple functions, their C++ code may be provided inline in the R console. Here is an exemplary function which takes a single numeric argument and returns a single numeric value:

```
Rcpp::cppFunction('
   double square(double x) {
      return x*x;
   }
')
```

Equivalently, a complete C++ source file may be written:

```
#include <Rcpp.h>
// [[Rcpp::plugins("cpp11")]]
using namespace Rcpp;

// [[Rcpp::export]]
double square(double x) {
   return x*x;
}
```

Usage in R:

```
square(2)
## [1] 4
```

Moreover, the following function takes a vector as input and returns a vector of the same size:

```
Rcpp::cppFunction('
   NumericVector square_vec(NumericVector x) {
      int n = x.size();
      NumericVector y(n);
      for (int i=0; i<n; ++i)
         y[i] = x[i]*x[i];
      return y;
   }
')

square_vec(c(-1, 2.5, 0))
## [1] 1.00 6.25 0.00
```

In this book we use R and C++ to implement the discussed algorithms. As a Python alternative to Rcpp, we suggest, e.g., Cython or boost::python.

Chapter 1

Aggregation of univariate data

Classically, the theory of aggregation discusses methods to summarize $n \geq 2$ numeric quantities in some real interval $\mathbb{I} = [a, b]$ or $]a, b[$, $a < b$. It is assumed that these quantities represent the results of *measurements* of the same process, for instance decision makers' preference degrees towards some alternative, or outputs gathered from sensors of the same kind (thermometers, traffic speed guns, personality questionnaires in psychology, and so forth). Of course, further on we shall discuss more complex methods, e.g., aggregating an arbitrary number of elements (so-called extended fusion functions), elements on discrete scales (nominal or ordered, like character strings), more complex objects (like vectors in $\mathbb{R}^d$ for $d > 1$ or DNA sequences), as well as determining numeric characteristics of entities. Before this happens, our universe of discourse appears to be quite simple at first glance, both from the mathematical and computational perspective. However, the kind reader should not be misled by that impression: the purpose of this introduction is not only to establish basic notation and key ideas. It shall turn out that even in such an uncomplicated domain a practitioner is faced with many challenges and interesting issues.

1.1 Preliminaries

To get a general idea of objects that are of our interest in this chapter, let us introduce the following definition.

Definition 1.1 ([93, 94]). A *fusion function* is a mapping $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$.

The notion of a fusion function reflects the abstract aim of data fusion: we take n numbers from some domain and, as a result result, get one value of the same *type*. For instance, in decision making and fuzzy logic we often suppose that $\mathbb{I} = [0, 1]$ or $\mathbb{I} = [-1, 1]$ and in statistics that $\mathbb{I} =]-\infty, \infty[$. We shall see

that the choice of interval $\mathbb{I} = [a, b]$ may be crucial; some of the results presented below may hold only if, e.g., $a = 0$ or $b = \infty$ but not otherwise.

Example 1.2. Consider a mapping defined as:

$$\mathsf{Sum}(\mathbf{x}) = \sum_{i=1}^{n} x_i.$$

It is a fusion function if, e.g., $\mathbb{I} = [0, \infty]$ or $[-\infty, \infty]$, but not if $\mathbb{I} = [0, 1]$ or $[-1, 1]$.

Let us review some general cases where fusion functions in $\mathbb{I}^n$ are applicable and introduce some well-known data aggregation tools.

Example 1.3. Assume that we are given a realization $(x_1, \ldots, x_n)$ of a random sample of independent random variables following a common distribution D with support $\mathbb{I}$. This may denote the results of an IQ test that was taken by a group of students. Knowing that D is symmetric around some value t, how can we estimate t so that one group of pupils may be compared to some reference value? Among examples of fusion functions applicable in this case we find:

— $\mathsf{AMean}(\mathbf{x}) = \dfrac{1}{n}\sum_{i=1}^{n} x_i,$ (*arithmetic mean*)

— $\mathsf{Median}(\mathbf{x}) = \begin{cases} x_{((n+1)/2)} & \text{if } n \text{ is odd,} \\ \left(x_{(n/2)} + x_{(n/2+1)}\right)/2 & \text{if } n \text{ is even.} \end{cases}$ (*median*)

Note that the sample median may be written as:

$$\mathsf{Median}(\mathbf{x}) = \frac{x_{\lfloor (n+1)/2 \rfloor} + x_{\lceil (n+1)/2 \rceil}}{2}$$

and that it is defined using order statistics, which are also types of fusion functions. Namely, for any $k \in [n]$, we may define:

$$\mathsf{OS}_k(\mathbf{x}) = x_{(k)}.$$

As we shall see in Section 1.7.3, the two following instances of order statistics are particularly noteworthy:

— $\mathsf{Min}(\mathbf{x}) = \mathsf{OS}_1(\mathbf{x}) = \bigwedge_{i=1}^{n} x_i,$ (*minimum*)

— $\mathsf{Max}(\mathbf{x}) = \mathsf{OS}_n(\mathbf{x}) = \bigvee_{i=1}^{n} x_i.$ (*maximum*)

Also, apart from the arithmetic mean, the reader is possibly familiar with two other types of means:

— $\mathsf{GMean}(\mathbf{x}) = \left(\prod_{i=1}^{n} x_i\right)^{1/n}$, (*geometric mean*)

— $\mathsf{HMean}(\mathbf{x}) = \frac{1}{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{x_i}}$. (*harmonic mean*)

It is well-known that for $a > 0$ we have:

$$\mathsf{Min}(\mathbf{x}) \leq \mathsf{HMean}(\mathbf{x}) \leq \mathsf{GMean}(\mathbf{x}) \leq \mathsf{AMean}(\mathbf{x}) \leq \mathsf{Max}(\mathbf{x}).$$

This is in fact one of the first results in the theory of aggregation – the Greek mathematicians studied its simplest case ($n = 2$) over 2000 years ago.

Example 1.4. Let us go back to Example 1.3. Knowing that some of the input observations were contaminated and that now outliers possibly occur in our data set (e.g., because the students were not focused enough while performing the tasks), how can we choose a fusion function F so that $\mathsf{F}(x_1, x_2, \ldots, x_n)$ is a plausible estimator of D's center point? Among possible choices we find:

— $\mathsf{TriMean}_k(\mathbf{x}) = \frac{1}{n-2k}\sum_{i=k+1}^{n-k} x_{(i)}$, (*trimmed mean*)

— $\mathsf{WinMean}_k(\mathbf{x}) = \frac{1}{n}\sum_{i=k+1}^{n-k} x_{(i)} + \frac{k}{n}x_{(k+1)} + \frac{k}{n}x_{(n-k)}$, (*Winsorized mean*)

for some $k \in \{0, 1, \ldots, \lfloor n/2 \rfloor - 1\}$.

Example 1.5 ([49]). Suppose that we have a rule-based system with rules of the form:

$$\texttt{IF } o_1 \texttt{ is } O_1 \texttt{ AND } o_2 \texttt{ is } O_2 \texttt{ AND ... AND } o_n \texttt{ is } O_n \texttt{ THEN...},$$

and that x_i denotes the degree of satisfaction of the predicate "o_i `is` O_i", $x_i \in [0, 1]$. At this point, 0 may be interpreted as "no satisfaction", 1 as "complete satisfaction", and intermediate values can depict partial degrees of compliance. Then the overall degree of satisfaction of all the rules may be referred to as $\mathsf{F}(x_1, x_2, \ldots, x_n)$. For the sake of this purpose the following fusion functions are sometimes used:

— $\mathsf{Prod}(\mathbf{x}) = \prod_{i=1}^{n} x_i$, (*product*)

— $\mathsf{Min}(\mathbf{x})$, (*sample minimum*)

— $\mathsf{T}_{\text{Ł}}(\mathbf{x}) = 0 \vee \left(\sum_{i=1}^{n} x_i - n + 1\right)$. (*Łukasiewicz t-norm*)

Example 1.6. Now let us assume that we have a rule-based system with rules of the form:

$$\texttt{IF } o_1 \texttt{ is } O_1 \texttt{ OR } o_2 \texttt{ is } O_2 \texttt{ OR } \ldots \texttt{ OR } o_n \texttt{ is } O_n \texttt{ THEN} \ldots.$$

Assuming that $x_i \in [0, 1]$, as in the previous example, we may take into account the fusion functions:

— $\mathsf{Max}(\mathbf{x})$, (*sample maximum*)

— $\mathsf{S}_{\text{Ł}}(\mathbf{x}) = 1 \wedge \left(\sum_{i=1}^{n} x_i\right)$, (*Łukasiewicz t-conorm*, *bounded sum*)

— $\mathsf{S}_{\text{D}}(\mathbf{x}) = \begin{cases} x_{(n)} & \text{if } x_{(n-1)} = 0, \\ 1 & \text{otherwise.} \end{cases}$ (*drastic t-conorm*)

Example 1.7. Similarly, in group decision making problems, x_i may designate the degree of preference of the ith expert towards an alternative. Here, F may be used to combine individual evaluations to obtain a global score, $\mathsf{F}(x_1, x_2, \ldots, x_n)$. Then, e.g., a bipolar scale (see [169] for discussion), $\mathbb{I} = [-1, 1]$, may be used, where -1 stands for "strongly disagree", and 1 for "strongly agree".

Additionally, suppose that the experts have different "esteem", i.e., some of them have stronger impact on the final decision than the others (this is the case of, e.g., stockholders in a company's board). Assuming that the ith expert is assigned weight $w_i \geq 0$, $\sum_{j=1}^{n} w_j = 1$, F is often set to be a convex combination of input values, that is:

— $\mathsf{WAMean}_{\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i$. (*weighted arithmetic mean*)

FP arithmetic. Before going any further let us make a remark concerning the representation of values in $\mathbb{I} \subseteq \mathbb{R}$ on modern computers.

Definition 1.8. For some $s, m \in \mathbb{N}$ and $\mathbb{N} \ni b \geq 2$ let:

$$\mathbb{F}^{b}_{s,m} = \left\{ \pm \sum_{i=0}^{s-1} d_i b^{j-i} : d_i \in [0 : b-1], j \in [-m : m] \right\} \subseteq \mathbb{R} \tag{1.1}$$

denote the set of signed *floating point numbers* with *precision* of s significant digits, *base* b, and *exponent* ranging in $\{-m, \ldots, m\}$.

In particular, if $b = 2$, then we have numbers in the binary representation (e.g., $1.0101_2 \cdot 2^4 = 21_{10}$), and if $b = 10$, then we get decimal numbers (e.g., $3.1415_{10} \cdot 10^0$). Equation (1.1) may be rewritten equivalently as:

$$\mathbb{F}^b_{s,m} = \left\{ \pm(d_0.d_1 d_2 \dots d_{s-1)_b} \cdot b^j, d_i \in [0 : b-1] : j \in [-m : m] \right\}.$$

In order to assure that each number $\mathbb{F}^b_{s,m}$ has an unambiguous representation, we may assume that $d_0 \neq 0$ (normalized form).

For some fixed b, s, m, let $\bar{\mathbb{F}}^b_{s,m} = \mathbb{F}^b_{s,m} \cup \{\pm\texttt{Inf}, \texttt{NaN}\}$, i.e., the set of extended floating point numbers that also includes signed infinities (representing values so small or so large that they do not fit in $\mathbb{F}^b_{s,m}$) and a not-a-number (an erroneous value, for results of operations like $\sqrt{-1}, \log(-1), 0/0 \notin \mathbb{R}$).

Definition 1.9. For fixed b, s, m, let $\mathsf{fp} : \mathbb{R} \to \bar{\mathbb{F}}^b_{s,m}$ be such that for arbitrary $x \in \mathbb{R}$ we have:

$$\mathsf{fp}(x) = \begin{cases} \texttt{Inf} & \text{if } x > \max \mathbb{F}^b_{s,m}, \\ -\texttt{Inf} & \text{if } x < \min \mathbb{F}^b_{s,m}, \\ \arg^* \min_{y \in \mathbb{F}^b_{s,m}} |x - y| & \text{otherwise.} \end{cases} \tag{1.2}$$

Thus, if $x \in \mathsf{range}(\mathbb{F}^b_{s,m})$, then $\mathsf{fp}(x)$ rounds x to the closest value in $\mathbb{F}^b_{s,m}$. Of course, such a rounding scheme may be ambiguous if b is even, e.g., in $\mathbb{F}^{10}_{3,2}$ the value $1{,}005_{10} \in \mathbb{R}$ can be represented as $1{,}00_{10}$ and $1{,}01_{10}$. In order for the function to be well defined, we should introduce some tie-breaking rule (and hence the informal notation $\arg^* \min$). Here we shall rely on the IEEE-754 standard which suggests the *round half to even* scheme, that is, d_{s-1} should always be even.

Definition 1.10. For fixed b, s, m, the *machine epsilon* is the greatest value $\varepsilon_{\mathrm{M}} > 0$ such that $\mathsf{fp}(1 + \varepsilon_{\mathrm{M}}) = 1$.

Proposition 1.11. *In any* $\mathbb{F}^b_{s,m}$ *it holds that* $\varepsilon_{\mathrm{M}} = b^{-s+1}/2$.

For example, in $\mathbb{F}^{10}_{3,2}$ we have $\varepsilon_{\mathrm{M}} = 5 \times 10^{-3} = 0.005$: it holds $\mathsf{fp}(1 + 0.005) = \mathsf{fp}(1.005) = 1$ as well as $\mathsf{fp}(1.005 + 0.0 \dots 1) = 1.01$.

The machine epsilon gives us an upper bound for the relative rounding error. This is because for each $0 \neq x \in \mathsf{range}(\mathbb{F})$ we have:

$$\left| \frac{x - \mathsf{fp}(x)}{x} \right| = \left| 1 - \frac{\mathsf{fp}(x)}{x} \right| \leq \varepsilon_{\mathrm{M}}.$$

Also, please note that there always exists $\delta \in [-\varepsilon_{\mathrm{M}}, \varepsilon_{\mathrm{M}}]$ such that $\mathsf{fp}(x) = x(1 + \delta)$.

It turns out that the `double` type, most often used for floating point computations on modern computers, is roughly equivalent to $\bar{\mathbb{F}}^2_{52/53,1023}$ according

to IEEE-754 (we omit issues concerning, among others, subnormal numbers). In this case, the machine epsilon is equal to 2^{-53}.

Remark 1.12. $\mathbb{F}_{s,m}^b$ is not closed with respect to the standard addition operation: $(\mathbb{F}_{s,m}^b, +)$ is not a subalgebra of $(\mathbb{R}, +)$. For instance, in $\mathbb{F}_{3,2}^{10}$ we have $0.001_{10} + 1_{10} = 1.001_{10} \notin \mathbb{F}_{3,2}^{10}$. In other words, even if two values are representable in $\mathbb{F}_{s,m}^b$ exactly, the result of applying "+" is might not necessarily be exact.

Let $\oplus : \bar{\mathbb{F}}_{s,m}^b \times \bar{\mathbb{F}}_{s,m}^b \to \bar{\mathbb{F}}_{s,m}^b$ be such that for $x, y \in \mathbb{R}$ and $\mathsf{fp}(\mathsf{fp}(x) + \mathsf{fp}(y)), \mathsf{fp}(x), \mathsf{fp}(y) \in \mathbb{F}_{s,m}^b$, it holds:

$$\mathsf{fp}(x) \oplus \mathsf{fp}(y) = \mathsf{fp}(\mathsf{fp}(x) + \mathsf{fp}(y)). \tag{1.3}$$

Moreover, let $\texttt{NaN} \oplus \tilde{z} = \tilde{z} \oplus \texttt{NaN} = \texttt{NaN}$ for $\tilde{z} \in \bar{\mathbb{F}}_{s,m}^b$, $\texttt{Inf} \oplus \tilde{z} = \tilde{z} \oplus \texttt{Inf} = \texttt{Inf}$, and $-\texttt{Inf} \oplus \tilde{z} = \tilde{z} \oplus -\texttt{Inf} = -\texttt{Inf}$ for $\tilde{z} \in \mathbb{F}_{s,m}^b$, as well as $\texttt{Inf} \oplus -\texttt{Inf} = -\texttt{Inf} \oplus \texttt{Inf} = \texttt{NaN}$. This is a typical redefinition of the ordinary addition operation so that it acts on elements in $\bar{\mathbb{F}}_{s,m}^b$. Other arithmetic operations, e.g., $\ominus, \otimes, \oslash$, may be introduced in a similar manner.

Remark 1.13. The $\oplus$ operation is not necessarily associative, i.e., for $\tilde{x}, \tilde{y}, \tilde{z} \in \bar{\mathbb{F}}_{s,m}^b$ we may have $(\tilde{x} \oplus \tilde{y}) \oplus \tilde{z} \neq \tilde{x} \oplus (\tilde{y} \oplus \tilde{z})$. For example, in $\bar{\mathbb{F}}_{3,2}^{10}$ it holds that:

$$\begin{aligned}
(0.005_{10} \oplus 0.025_{10}) \oplus 1.00_{10} &= \\
0.03_{10} \oplus 1.00_{10} &= \\
1.03_{10} &\neq 0.005_{10} \oplus (0.025_{10} \oplus 1.00_{10}) \\
&= 0.005_{10} \oplus 1.02_{10} \\
&= 1.02_{10}.
\end{aligned}$$

Remark 1.14. Let us study the absolute error of the $\oplus$ operation. Let $\tilde{x} \oplus \tilde{y} \oplus \tilde{z} = (\tilde{x} \oplus \tilde{y}) \oplus \tilde{z}$, where $0 < \tilde{x}, \tilde{y}, \tilde{z} \in \mathbb{F}_{s,m}^b$ and $\tilde{x} \oplus \tilde{y} \oplus \tilde{z} \in \mathbb{F}_{s,m}^b$. For some $\delta_1, \delta_2 \in [-\varepsilon_{\mathrm{M}}, \varepsilon_{\mathrm{M}}]$ we have:

$$\begin{aligned}
\tilde{x} \oplus \tilde{y} \oplus \tilde{z} - (\tilde{x} + \tilde{y} + \tilde{z}) &= \mathsf{fp}(\mathsf{fp}(\tilde{x} + \tilde{y}) + \tilde{z}) - (\tilde{x} + \tilde{y} + \tilde{z}) \\
&= \delta_1 \tilde{x} + \delta_1 \tilde{y} + \delta_2(\tilde{x} + \tilde{y} + \delta_1 \tilde{x} + \delta_1 \tilde{y} + \tilde{z}) \\
&\leq \varepsilon_{\mathrm{M}}((2 + \varepsilon_{\mathrm{M}})(\tilde{x} + \tilde{y}) + \tilde{z}).
\end{aligned}$$

The relative error of $\oplus$ is not greater than $\varepsilon_{\mathrm{M}}(1 + (\tilde{x} + \tilde{y})(1 + \varepsilon_{\mathrm{M}}))/(\tilde{x} + \tilde{y} + \tilde{z})$. However, we observe that this upper bound depends on the relative magnitude of the inputs: $\tilde{z} \geq \tilde{x}$ and $\tilde{z} \geq \tilde{y}$ leads to the largest error. From that we may imply that, e.g., the Sum fusion function imposes the smallest relative error if we add up nonnegative values in an increasing order.

We see that even though the fusion functions studied so far seemed to be very uncomplicated, special care should be taken when they are implemented on

a computer. In such a setting, they of course are mappings like $\bar{\mathsf{F}} : \bar{\mathbb{I}}^n \to \bar{\mathbb{I}}$, where $\bar{\mathbb{I}} = [a, b] \cap \bar{\mathbb{F}}^b_{s,m}$, $a, b \in \bar{\mathbb{F}}^b_{s,m}$. For the sole Sum function there exists a number of algorithms; one of them is the Kahan (compensated) summation routine [266], see also [247].

Remark 1.15. The $\oplus$ and $\otimes$ operations are not distributive in general: For example in $\bar{\mathbb{F}}^{10}_{3,2}$ we have:

$$\begin{aligned}
(0.001_{10} \otimes 0.1_{10}) \oplus (0.001_{10} \otimes 1.00_{10}) &= \\
0.000_{10} \oplus 0.001_{10} &= \\
0.001_{10} &\neq 0.001_{10} \otimes (0.1_{10} \oplus 1.00_{10}) \\
&= 0.001_{10} \otimes 0.1_{10} \\
&= 0.000_{10}.
\end{aligned}$$

The reader is suggested to refer, e.g., to [282, Section 4.2], [246], or [226] for further discussion and issues on the topic.

Remark 1.16. There are a few libraries for performing floating point computations with higher precision, for instance MPFR (Multiple Precision Floating-Point Reliable) and GMP (GNU Multiple Precision) libraries. Unfortunately, the errors in numerical computations are inherent, they may only be reduced. This of course comes at a cost of slowing down the computations.

As an alternative, one may consider computer algebra systems performing *symbolic* computations, e.g., Mathematica, Maxima, Maple, or Sage.

Example 1.17. The only discussed so far fusion functions $\bar{\mathsf{F}} : \bar{\mathbb{I}}^n \to \bar{\mathbb{I}}$ that produce exact values are OS_k for arbitrary $k \in [n]$ (but not Median for arbitrary n) and S_{D}. Generally, among precise operations in $\bar{\mathbb{F}}$ we find $\wedge$ and $\vee$, which are the basis for the class of weighted lattice polynomial functions discussed in Section 1.7.5, see also Equation (1.32).

1.2 Properties of fusion functions

The definition of a fusion function we presented above is very general. Thus, we would like to narrow it down and identify some crucial properties that must always be fulfilled in order to say that F might at least be potentially interesting to us. This, however, is relative to the nature of the practical problem we are faced with.

In the following subsections we therefore explore some noteworthy frameworks, which include nondecreasingness, symmetry, idempotence, different types of equivariances, additivity and so forth.

1.2.1 Nondecreasingness and preservation of end points

In Examples 1.5, 1.6, and 1.7 it seems that it is reasonable to require that if we increase the degree of satisfaction of a predicate or the degree of preference stated by the ith expert, then the new overall valuation should not be smaller than the previous one. Such a property may be formalized as follows. Let $\leq_n$ be a binary relation on $\mathbb{I}^n$ such that $\mathbf{x} \leq_n \mathbf{y}$ if for all $i \in [n]$ we have $x_i \leq y_i$.

Definition 1.18. A fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is called *nondecreasing* (in each variable), whenever for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$ it holds that if $\mathbf{x} \leq_n \mathbf{y}$, then $\mathsf{F}(\mathbf{x}) \leq \mathsf{F}(\mathbf{y})$.

Remark 1.19. All the fusion functions reviewed so far are nondecreasing.

We may also define a *strictly increasing* function by assuming that $\mathbf{x} <_n \mathbf{y} \Rightarrow \mathsf{F}(\mathbf{x}) < \mathsf{F}(\mathbf{y})$, where $\mathbf{x} <_n \mathbf{y}$ if and only if $\mathbf{x} \leq_n \mathbf{y}$ and $\mathbf{x} \neq \mathbf{y}$. Moreover, *unanimous increasingness* (compare [230]), also known as joint strict monotonicity, can be defined by considering the cases in which for all $i \in [n]$ it holds $x_i < y_i$.

Example 1.20. Among strictly increasing fusion functions we find, e.g., AMean. Moreover, Min and Max are unanimously increasing.

Moreover, we may require that F should at least be normalized in such a way that it preserves the endpoints of $\mathbb{I}$.

Definition 1.21. We say that a fusion function F is *endpoint-preserving*, whenever it holds that $\mathsf{F}(n * a) = a$ and $\mathsf{F}(n * b) = b$.

In other words, e.g., in a decision making problem, if the criteria are not satisfied at all or each expert finds an alternative totally plausible, then in such extreme cases the result should be concordant with the inputs. Note that if $\mathbb{I}$ is an open interval, then by, e.g., $\mathsf{F}(n * a)$ would of course mean $\mathsf{F}(n * a) = \lim_{\mathbf{x} \to (n*a)} \mathsf{F}(\mathbf{x})$.

Example 1.22. Prod is an endpoint-preserving fusion function for input elements in $[0, 1]$, but not when $\mathbb{I} = [0, 0.5]$ or $\mathbb{I} = [-1, 1]$ and even n is considered. Moreover, it is not nondecreasing, e.g., in the $[-1, 1]$ case. Hence, we see that some properties indeed depend on the choice of $\mathbb{I} = [a, b]$ as well as n.

On the other hand, $\mathsf{F}(\mathbf{x}) = b \wedge \mathsf{Prod}(\mathbf{x})$ is endpoint-preserving in the case $\mathbb{I} = [0, b]$ for any $b \geq 1$. We shall often observe that some fusion functions may be "tuned up": by applying particular transformations they start to fulfill a given property which is of interest in a particular domain.

Properties given in Definitions 1.18 and 1.21 lead us to the classical definition of an aggregation function, as in [39, 49, 230].

Definition 1.23. $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is an *aggregation function* whenever it is nondecreasing in each variable and it is endpoint-preserving.

Note that if F is nondecreasing, then it is endpoint-preserving if and only if $\inf_{\mathbf{x}\in\mathbb{I}^n} \mathsf{F}(\mathbf{x}) = a$ and $\sup_{\mathbf{x}\in\mathbb{I}^n} \mathsf{F}(\mathbf{x}) = b$.

1.2.2 Idempotence and internality

From another standpoint, fusion functions to be used in application domains like those mentioned in Examples 1.3 and 1.4, might not necessarily be nondecreasing. For example, already Kolmogorov and Nagumo [292, 370], compare also Aczel's paper [4], were interested in discussing various kinds of *means*. One of the properties they required is the so-called idempotency (unanimity, compensativity), which is well-known from algebra, where we say that element x is idempotent with respect to a binary operator $*$, if we have $x * x = x$. The following definition extends this property to n-ary aggregation functions, see [230].

Definition 1.24. A fusion function F is called *idempotent*, whenever:

$$(\forall x \in \mathbb{I})\ \mathsf{F}(n * x) = x. \tag{1.4}$$

Intuitively, if we aggregate n equal inputs, the resulting value should fully agree with them. Note that each idempotent fusion function is also endpoint-preserving. Among idempotent aggregation functions we find WAMean and OS_k, but not T_{L} and S_{D}.

Remark 1.25. Let $n = 10$, $\overline{\mathsf{HMean}}(\mathbf{x}) = n \oslash ((1 \oslash x_1) \oplus \cdots \oplus (1 \oslash x_n))$ (the floating point equivalent to the harmonic mean with respect to the `double` type), and x be equal to:

$$+1.1101110110000111101111001100111111010001110011110111 1_2 \times 2^0,$$

that is $x \simeq 1.865352440541410805608_{10}$. Then $\overline{\mathsf{HMean}}(n * x)$ is equal to:

$$+1.1101110110000111101111001100111111010001110011111000 0_2 \times 2^0,$$

i.e., $x \neq \overline{\mathsf{HMean}}(n*x) \simeq 1.8653524405414110_{10}$. Even if – algebraically – HMean is idempotent, its "computer version" is not. Therefore, one should be careful when comparing results of floating point computations, especially with the $=$ operator. A much more reliable way to do so is to test whether $|\mathsf{F}(n*x)-x|/|x| \leq \varepsilon$ for some small ε being a function of the machine epsilon, e.g., $\varepsilon = \sqrt{\varepsilon_{\mathrm{M}}}$.

Another significant property – internality (as named in [230, Definition 2.53]), also known as compensativity – requires that a fusion function's output value

must lie "somewhere in-between" the input values (see page 106 for an alternative setting).

Definition 1.26. A fusion function F is *internal* whenever $(\forall \mathbf{x} \in \mathbb{I}^n)$ we have:

$$\mathsf{Min}(\mathbf{x}) \leq \mathsf{F}(\mathbf{x}) \leq \mathsf{Max}(\mathbf{x}). \tag{1.5}$$

In other words, $\mathsf{F}(\mathbf{x}) \in \operatorname{intcl} \mathbf{x}$.

We see that each internal F is idempotent too. We often consider fusion functions which are both idempotent and nondecreasing. Actually, idempotent aggregation functions are sometimes called *averaging functions* in the literature, compare [230]. It turns out that in such a case idempotency and internality coincide, see [230, Proposition 2.54] for the proof.

Proposition 1.27. *If* F *if nondecreasing and idempotent, then it is internal.*

Please observe that nondecreasingness is appealing from the mathematical perspective: it turns out that many other properties can be simplified owing to this property. Though, in some applications it may not be fully desirable.

Remark 1.28. According to [230], already Cauchy in 1821 considered under a name *mean* an internal, but not necessarily nondecreasing fusion function. Similarly, compare [39], Gini in the 1950s required only this very property when discussing various means.

Remark 1.29. A class of idempotent, but not necessarily nondecreasing fusion functions may be useful in the case of aggregating data in the presence of outliers: we might sometimes want to allow that $F(0,0,\ldots,0,1) > F(0,0,\ldots,0,10^9)$, just as in Example 1.4. For instance, it is not uncommon to define outliers (e.g., when building box-and-whisker plots) as observations x_i such that $x_i < \mathsf{Q}_{0.25}(\mathbf{x}) - 1.5(\mathsf{Q}_{0.75}(\mathbf{x}) - \mathsf{Q}_{0.25}(\mathbf{x}))$ or $x_i > \mathsf{Q}_{0.75}(\mathbf{x}) + 1.5(\mathsf{Q}_{0.75}(\mathbf{x}) - \mathsf{Q}_{0.25}(\mathbf{x}))$, where $\mathsf{Q}_{0.25}$ and $\mathsf{Q}_{0.75}$ stand for the 1st and the 3rd quartile, compare Example 1.83. Then, having F defined as "the arithmetic mean of all non-outlying observations", we get, e.g., $0.4 = \mathsf{F}(-2,-1,0,1,4) \not\leq \mathsf{F}(-2,-1,0,1,5) = -0.5$. In Section 1.2.8 we shall make a review of other types of monotonicities.

Remark 1.30. The *mode*, well known in exploratory data analysis, is defined as an observation that appears most often in the input data set. Of course, such a definition is not strict in the case of multimodal vectors. What is important here, however, is that it is an idempotent yet not monotone (at least with respect to $\leq_n$) fusion function. It is because we have, e.g., $3 = \mathsf{Mode}(1,1,2,2,3,3,3) < \mathsf{Mode}(2,2,2,2,3,3,3) = 2$.

1.2.3 Conjunctivity and disjunctivity

One may well ask if also idempotence or internality might not be desirable in certain contexts. The answer is of course positive.

Remark 1.31. In an AND-based rule aggregation system from Example 1.5, small values of x_i may be treated as "noise" and may be cut down by F to 0. What is more, in Example 1.6 quite an opposite fusion functions' behavior is expected.

Actually, Dubois and Prade (see [165, 166, 169]) propose to distinguish the following four main classes of fusion functions:

— internal (averaging),

— conjunctive (AND-like, e.g., t-norms),

— disjunctive (OR-like, e.g., t-conorms),

— mixed.

This distinction is based on the relationship between these functions and Min or Max.

Definition 1.32. A fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is *conjunctive*, whenever for all $\mathbf{x} \in \mathbb{I}^n$ we have:

$$\mathsf{F}(\mathbf{x}) \leq \mathsf{Min}(\mathbf{x}). \tag{1.6}$$

Definition 1.33. A fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is called *disjunctive*, whenever for every $\mathbf{x} \in \mathbb{I}^n$ it holds:

$$\mathsf{Max}(\mathbf{x}) \leq \mathsf{F}(\mathbf{x}). \tag{1.7}$$

Note that Min and Max are internal as well as conjunctive and disjunctive, respectively, at the same time. On the other hand, mixed fusion functions are neither internal, conjunctive, nor disjunctive (for all input vectors). We shall see in Section 1.5.3 that this is the case of uninorms (among others).

Example 1.34. Assuming that $\mathbb{I} = [0, 1]$, the so-called *3-Π* function, given by:

$$3\Pi(\mathbf{x}) = \frac{\mathsf{Prod}(\mathbf{x})}{\mathsf{Prod}(\mathbf{x}) + \mathsf{Prod}(1 - \mathbf{x})},$$

is an example of a mixed-type fusion function, with convention $0/0 = 0$. Yager and Rybalov [483] showed that it is conjunctive on $[0, 0.5]^n$, disjunctive on $[0.5, 1]^n$, and internal otherwise.

Even though the focus of this book is generally on functions that are idempotent, mappings from other classes are anyway noteworthy, have influenced, and

continue to be a very important part of the theory of aggregation. For instance, the notion of a copula (a conjunctive – among others – fusion function, e.g., Min or T_{L}) will be useful in Chapter 2 when we discuss various methods for generating random observations from $\mathbb{R}^d$ for $d > 1$. Hence, from time to time, we shall refer back to them.

1.2.4 Symmetry. Permutations of inputs

Another useful property is called symmetry. It may be a sine qua non condition in statistics, where all the observations are treated just as "points in the real line". Moreover, it may be useful in decision making, in a case when all of the experts are of the same "esteem" or all of them are anonymous.

Definition 1.35. We say that a fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is *symmetric*, if:

$$(\forall \mathbf{x}, \mathbf{y} \in \mathbb{I}^n)\ \mathbf{x} \cong \mathbf{y} \Longrightarrow \mathsf{F}(\mathbf{x}) = \mathsf{F}(\mathbf{y}), \tag{1.8}$$

where $\mathbf{x} \cong \mathbf{y}$ if and only if there exists a permutation σ of $[n]$ such that $\mathbf{x} = (y_{\sigma(1)}, \ldots, y_{\sigma(n)})$.

In other words, the output value of a symmetric function does not depend on the ordering of inputs. Of course, each F that is defined as a function of $x_{(1)}, \ldots, x_{(n)}$, i.e., order statistics, is symmetric by definition, see also Section 1.3.1.C.

Example 1.36. Among instances of symmetric aggregation functions we find the sample median, all order statistics, or trimmed and Winsorized means (see Example 1.4). Specifically, the 1-trimmed mean is used in ski jumping competitions organized by the International Ski Federation, where each of 5 experts provide scores based on a jumper's balance, body position, and landing style. In such a case, one lowest and highest score is neglected.

Remark 1.37. For a given k, $x_{(k)}$ may be computed in $O(n)$ time by using the BFPRT (median of medians, [66]) algorithm without actually sorting the input vector. Note that often the Quickselect [250] or the Floyd-Rivest [194] schemes are preferred, though; they have $O(n)$ time complexity only on average but, when implemented, tend to run faster than BFPRT. See also `std::nth_element()` function in the C++ Standard Library.

Remark 1.38 ([38]). As $\mathsf{WinMean}_k(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\big((x_i \vee x_{(k+1)}) \wedge x_{(n-k)}\big) = \frac{1}{n}\sum_{i=1}^{n}\mathsf{Median}(x_{(k+1)}, x_i, x_{(n-k)})$, the computation of a Winsorized mean has $O(n)$ time complexity. Moreover, the same holds for a trimmed mean, because:

$$\mathsf{TriMean}_k(\mathbf{x}) = \frac{1}{n-2k}\left(n\mathsf{WinMean}_\alpha(\mathbf{x}) - kx_{(n-k)} - kx_{(k+1)}\right).$$

Another interesting result concerning algorithmic aspects of symmetric fusion functions is due to J. Rotman, see [230, Proposition 2.33]. It states that we need to compute the value of F only three times in order to determine if this property holds for a fixed $\mathbf{x}$.

Proposition 1.39. *A fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is symmetric if and only if for all $\mathbf{x} \in \mathbb{I}^n$:*

$$\begin{aligned} \mathsf{F}(x_1, x_2, x_3, \ldots, x_{n-1}, x_n) &= \mathsf{F}(x_2, x_1, x_3, \ldots, x_{n-1}, x_n) \text{ and} \\ \mathsf{F}(x_1, x_2, x_3, \ldots, x_{n-1}, x_n) &= \mathsf{F}(x_2, x_3, x_4, \ldots, x_n \quad, x_1) \end{aligned}$$

Permutations of input objects play an important role in data fusion theory. Thus, let us recall an algorithm for generating a random permutation of a given vector. By "random" we of course mean a situation in which every possible permutation is assigned the same probability measure, i.e., the distribution is uniform. Generating a random permutation is not necessarily straightforward: in particular, a procedure like "n times swap two randomly selected elements of $\mathbf{x}$" does not lead to a uniform distribution. To achieve this goal, we should rather rely, e.g., on the following algorithm.

Algorithm 1.40. *Generation of a random permutation of elements of a given vector $\mathbf{x} = (x_1, \ldots, x_n)$ with the Fisher-Yates shuffle [193] as formulated by Knuth [282, Algorithm P] is done as follows:*

1. *Let σ be such that $\sigma(i) = i$ for all $i \in [n]$;*
2. *For $j = n, n-1, \ldots, 2$ do:*
   - 2.1. *Let i be a random number uniformly distributed in $\{1, 2, \ldots, j\}$;*
   - 2.2. *Swap $\sigma(i) \leftrightarrow \sigma(j)$;*
3. *Return $\mathbf{x}_\sigma$.*

It is easily seen that the above procedure runs in $O(n)$ time. Moreover, note that with Algorithm 1.40 we do not only get a random rearrangement of elements of $\mathbf{x}$ but also a random $\sigma \in \mathfrak{S}_{[n]}$ itself.

Moreover, later on we need the notion of comonotonicity. As it is somehow related to the topics discussed in this subsection, let us introduce it now.

Definition 1.41. According to [230, Definition 2.123], $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$ are *comonotonic*, denoted by $\mathbf{x} \Cap \mathbf{y}$, if and only if there exists a permutation $\sigma \in \mathfrak{S}_{[n]}$ such that:

$$x_{\sigma(1)} \leq \cdots \leq x_{\sigma(n)} \quad \text{and} \quad y_{\sigma(1)} \leq \cdots \leq y_{\sigma(n)}. \tag{1.9}$$

Thus, σ orders $\mathbf{x}$ and $\mathbf{y}$ simultaneously. It is easily seen that the $\Cap$ binary relation is reflexive and symmetric.

Remark 1.42. Equivalently, $\mathbf{x}$ and $\mathbf{y}$ are comonotonic, if and only if for every $i, j \in [n]$ it holds that:

$$(x_i - x_j)(y_i - y_j) \geq 0.$$

It is easily seen that in order to generate two random comonotonic vectors $\mathbf{x}, \mathbf{y}$ we may generate the two vectors independently (from some desired probability distribution on $\mathbb{I}^n$), sort them separately, generate a random permutation σ with Algorithm 1.40, and then return $(\mathbf{x}_\sigma, \mathbf{y}_\sigma)$.

If all the elements of $\mathbf{x}$ are unique, then to determine if two vectors are comonotonic it is sufficient to take the (unique) ordering permutation of $\mathbf{x}$ and then verify if $\mathbf{y}_\sigma$ is sorted. On the other hand, if there are tied observations in $\mathbf{x}$, we seek the longest possible sequence $(x_{\sigma(i)}, x_{\sigma(i+1)}, \ldots, x_{\sigma(i+k)})$ such that $x_{\sigma(i)} = x_{\sigma(i+k)}$, where σ is an ordering permutation of $\mathbf{x}$. Then we try to update σ so that it also sorts the corresponding observations in $\mathbf{y}$, see [207] for discussion. An exemplary implementation is given in Figure A.1. Here we use a sorting routine from the C++11 Standard Library, with guaranteed run-time of $O(n \log n)$. Note that the provided implementation also generates a common ordering permutation for $\mathbf{x}$ and $\mathbf{y}$. What is more, at some step an ordering permutation of $\mathbf{x}$ is given. To guarantee that σ is unique and such that for $i < j$ and $x_i = x_j$ we have $\sigma(i) \leq \sigma(j)$, one may use `std::stable_sort()` instead of `std::sort()`.

1.2.5 Continuity and convexity

The notion of continuity is very attractive from the perspective of mathematical analysis.

Definition 1.43. $\mathsf{F} : \mathbb{I}^n \to \mathbb{R}$ is *continuous* if for all $\mathbf{x}^* \in \mathbb{I}^n$ we have:

$$\lim_{\mathbb{I}^n \ni \mathbf{x} \to \mathbf{x}^*} \mathsf{F}(\mathbf{x}) = \mathsf{F}(\mathbf{x}^*). \tag{1.10}$$

It may be shown that if F is nondecreasing, then F is continuous if and only if it is continuous in each variable, see [230, Proposition 2.8] for a proof. This corresponds to the so-called *intermediate value property*: for each $\mathbf{x}, \mathbf{y}$ with $\mathbf{x} \leq_n \mathbf{y}$ and $c \in [\mathsf{F}(\mathbf{x}), \mathsf{F}(\mathbf{y})]$ there exists $\mathbf{z}$ such that $\mathsf{F}(\mathbf{z}) = c$. In fact, $\mathbf{z} = \alpha\mathbf{x} + (1-\alpha)\mathbf{y}$ for some $\alpha \in [0, 1]$, i.e., it is a convex combination of $\mathbf{x}$ and $\mathbf{y}$.

Remark 1.44. A *mean* in the sense of Kolmogorov [292] and Nagumo [370] is a fusion function F that is nondecreasing, continuous, symmetric, and idempotent.

Let us recall the definition of a norm on an abstract vector space V over a subfield $\mathbb{R}$.

Definition 1.45. A *norm* on V is a function $\|\cdot\| : V \to [0, \infty]$ such that:

(a) $\|\mathbf{v}\| = 0$ if and only if $\mathbf{v} \equiv \mathbf{0}$,

(b) for any $a \in \mathbb{R}$ and $\mathbf{v} \in V$ it holds $\|a\mathbf{v}\| = |a| \, \|\mathbf{v}\|$ (homogeneity of the first degree), and

(c) for all $\mathbf{u}, \mathbf{v} \in V$ we have $\|\mathbf{u} + \mathbf{v}\| \leq \|\mathbf{u}\| + \|\mathbf{v}\|$ (*triangle inequality*).

Moreover, we call $\|\cdot\|$ a *pseudonorm* if condition (a) is replaced with:

(a') $\|\mathbf{v}\| = 0$ if $\mathbf{v} \equiv \mathbf{0}$.

Informally, a norm is often used to measure the "size" of an object and is a type of its numeric characteristic, compare Chapter 5. Also, we further on recall that norms may be used to generate various distance metrics, that is functions to measure dissimilarities between pairs of objects. It shall turn out that such a notion is meaningful in aggregation theory (and data fusion and mining), as many fusion functions may be written as minimizers of some penalty function.

Remark 1.46. Here are some notable norms on $\mathbb{R}^n$:

— *Euclidean norm*, $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2} = \sqrt{\mathbf{x}^T \mathbf{x}}$,

— *Manhattan (Taxicab) norm*, $\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$,

— *maximum (Chebyshev) norm*, $\|\mathbf{x}\|_\infty = \bigvee_{i=1}^{n} |x_i|$,

or, more generally:

— *p-(Minkowski-)norm*, $p \geq 1$, $\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$.

R. Lipschitz [322] also considered the following condition.

Definition 1.47. We say that a fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is *Lipschitz continuous* if for any norm $||\cdot||$ on $\mathbb{I}^n$ there exists a finite constant $K \geq 0$ such that for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$ it holds:

$$|\mathsf{F}(\mathbf{x}) - \mathsf{F}(\mathbf{y})| \leq K||\mathbf{x} - \mathbf{y}||. \tag{1.11}$$

Of course, K depends on the choice of the norm. If no norm is mentioned explicitly, the Manhattan one is assumed. In such a case, the smallest constant in the above equation such that the Lipschitz condition is still fulfilled is called the (best) Lipschitz constant. In particular, we call a function *1-Lipschitz* if:

$$|\mathsf{F}(\mathbf{x}) - \mathsf{F}(\mathbf{y})| \leq \sum_{i=1}^{n} |x_i - y_i|.$$

For example, this is the case of the arithmetic mean, with best K being equal to $1/n$ in the case of the 1-norm.

Generally, if the best K is not greater than 1 we call F *non-expansive*, and if $K < 1$ then F is a *contraction*.

It might be shown that if F is a Lipschitz function, then it is continuous. On the other hand, GMean is an example of a continuous aggregation function on $[0, \infty[^n$ which is not Lipschitz.

Further on we shall see that, e.g., copulas, widely used in probability and mathematical modeling (hydrology, finance, risk, etc.) are Lipschitz functions. In this regard, note that a continuous fusion function F acting on a list with elements in a real closed interval fulfills the property that for each ε, there exists $\delta > 0$ such that $|\mathsf{F}(\mathbf{x}) - \mathsf{F}(\mathbf{y})| \leq \varepsilon$ if $\mathbf{x}, \mathbf{y}$ are such that $\|\mathbf{x} - \mathbf{y}\| \leq \delta$.

Remark 1.48. As the set of floating point numbers $\mathbb{F}$ is countable, the notion of continuity is rather of theoretical interest. The Lipschitz condition is even stronger: it guarantees that an arbitrarily small change of input elements does not lead to uncontrolled behavior of the output. Nevertheless, its milder version is useful from the computational perspective, where it is called *numerical stability* and concerns "small" perturbations of values in $\mathbf{x}$.

The next property is important, e.g., when dealing with optimization tasks.

Definition 1.49. We say that a fusion function F is *convex* whenever for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$ and $\lambda \in [0, 1]$ it holds that:

$$\mathsf{F}(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda\mathsf{F}(\mathbf{x}) + (1-\lambda)\mathsf{F}(\mathbf{y}). \tag{1.12}$$

Moreover, F is called *concave*, if $(b + a) - \mathsf{F}$ is convex.

Example 1.50. Max is an example of a convex function and GMean is a concave one if $a > 0$. WAMean is both convex and concave at the same time. Moreover, by definition, every norm on $\mathbb{R}^n$ is convex.

Note that if F is continuous and twice differentiable, then it is convex if and only if its Hessian is positive semidefinite. Also, if F and G are convex fusion functions, then all of their convex combinations ($c\mathsf{F} + d\mathsf{G}$ for any $c, d \geq 0$, $c + d = 1$) are also convex.

1.2.6 Equivariance to translation and scaling

In the practice of data analysis, transformations of input variables such as *standardization*:

$$\mathbf{x} \mapsto \frac{\mathbf{x} - \mathsf{AMean}(\mathbf{x})}{\mathsf{SD}(\mathbf{x})},$$

robust standardization:

$$\mathbf{x} \mapsto \frac{\mathbf{x} - \mathsf{Median}(\mathbf{x})}{\mathsf{MAD}(\mathbf{x})},$$

or *normalization*:

$$\mathbf{x} \mapsto \frac{\mathbf{x} - \mathsf{Min}(\mathbf{x})}{\mathsf{Range}(\mathbf{x})},$$

where $0/0 = 0$, are often applied. Here $\mathsf{SD}(\mathbf{x}) = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mathsf{AMean}(\mathbf{x}))^2}$ is the sample *standard deviation*, $\mathsf{MAD}(\mathbf{x}) = 1.4826\mathsf{Median}(|\mathbf{x} - \mathsf{Median}(\mathbf{x})|)$ is the *median absolute deviation*, and $\mathsf{Range}(\mathbf{x}) = \mathsf{Max}(\mathbf{x}) - \mathsf{Min}(\mathbf{x})$ is the *range*, see also Section 5.2.

The classic standardization implies that the transformed vector is of mean 0 and standard deviation of 1 and normalization assures that the output values are in $[0, 1]$. The three transformations retain relative distances between the observations.

Additionally, it is not that uncommon to convert the measurement units (e.g., Fahrenheit to Celsius, feet to meters, etc.). Note that standardization and normalization result in *unitless* values.

Taking the above into account, sometimes it would be useful to assure that a fusion function is equivariant to translation (shifting) and/or scaling.

Definition 1.51. We say that a fusion function F is *translation (shift, difference scale) equivariant* if for all $t \in \mathbb{R}$ and $\mathbf{x} \in \mathbb{I}^n$ such that $t + \mathbf{x} \in \mathbb{I}^n$ it holds:

$$\mathsf{F}(t + \mathbf{x}) = t + \mathsf{F}(\mathbf{x}). \tag{1.13}$$

Note that *translation invariance* would imply that for all t and $\mathbf{x}$ it held $\mathsf{F}(t + \mathbf{x}) = \mathsf{F}(\mathbf{x})$.

Remark 1.52. Notably, Bullen in his seminal monograph [87] defines a *mean* as a nondecreasing, symmetric, idempotent, and translation equivariant fusion function. Interestingly, he assumes that means are most often computed on elements in the interval $\mathbb{I} = [0, \infty[$.

Definition 1.53. A fusion function F is called *(ratio) scale equivariant* if for all $s > 0$ and $\mathbf{x} \in \mathbb{I}^n$ such that $s\mathbf{x} \in \mathbb{I}^n$ it holds that:

$$\mathsf{F}(s\mathbf{x}) = s\mathsf{F}(\mathbf{x}). \tag{1.14}$$

Similarly to the translation or scale equivariance, $\wedge$- and $\vee$-*equivariance* may be defined: it suffices to replace the addition or multiplication operation with the minimum and maximum, respectively. In this regard, translation and scale equivariance may be combined as follows.

Definition 1.54. A fusion function F is *interval scale equivariant* if for all $s > 0$, $t \in \mathbb{R}$, $\mathbf{x} \in \mathbb{I}^n$ with $t + s\mathbf{x} \in \mathbb{I}^n$ we have:

$$\mathsf{F}(t + s\mathbf{x}) = t + s\mathsf{F}(\mathbf{x}). \tag{1.15}$$

Remark 1.55. Note that if $0 \in \mathbb{I}$ and F is at least scale equivariant, then for every s we have that $s\mathsf{F}(\mathbf{0}) = \mathsf{F}(s\mathbf{0}) = \mathsf{F}(\mathbf{0})$. Thus, $\mathsf{F}(\mathbf{0}) = 0$. If F is additionally translation equivariant, then for any t we have that $\mathsf{F}(n{*}t) = \mathsf{F}(t\mathbf{1}) = t\mathsf{F}(\mathbf{0}{+}1) = t$. Thus, F is idempotent.

Remark 1.56. It is worth noting that Pitman in 1939 [392] considered *estimators* of a location parameter $l \in \mathbb{R}$ under the transformation:

$$f(x) \mapsto \frac{1}{s} f\left(\frac{x - l}{s}\right)$$

of a density function f with $s > 0$. We see that it is a simple translate-scale model. He posed that $\mathsf{A} : \mathbb{R}^n \to \mathbb{R}$, being the estimator of l, should fulfill:

$$\mathsf{A}\left(\frac{x_1 + \lambda}{\mu}, \ldots, \frac{x_n + \lambda}{\mu}\right) = \frac{\mathsf{A}(x_1, \ldots, x_n) + \lambda}{\mu}$$

for all $\lambda \in \mathbb{R}$ and $\mu > 0$, and be independent of l. The definition of the A function is a very appealing, early approach to aggregation as we know today: "any function of this type will be called an estimate of l", see [392, page 409]. He also wrote on page 420: "any function of the sample values whose value may be used as an estimate of an unknown parameter is called an estimator of that parameter". Moreover, he pointed out that there are many estimators, each of which may fulfill different properties (e.g., one that minimizes the minimum mean absolute error or the minimum mean square error).

Sometimes we might be interested in the following, much stronger version of interval scale equivariance (compare Proposition 1.67):

Definition 1.57. A fusion function F is said to be *ordinal scale equivariant* if for all increasing bijections $\varphi : \mathbb{I} \to \mathbb{I}$ and every $\mathbf{x} \in \mathbb{I}^n$ it holds that:

$$\mathsf{F}(\varphi(x_1), \ldots, \varphi(x_n)) = \varphi(\mathsf{F}(\mathbf{x})). \tag{1.16}$$

1.2.7 Additivity

Recall that the "+", "∧", and "∨" operations on vectors are applied elementwise.

Definition 1.58. A fusion function F is said to be *additive*, whenever:

$$\mathsf{F}(\mathbf{x}+\mathbf{y}) = \mathsf{F}(\mathbf{x}) + \mathsf{F}(\mathbf{y}), \tag{1.17}$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$ such that $\mathbf{x}+\mathbf{y} \in \mathbb{I}^n$.

It is easily seen that each idempotent and additive fusion function is also translation equivariant.

Definition 1.59. A fusion function F is said to be *modular*, whenever for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$:

$$\mathsf{F}(\mathbf{x} \wedge \mathbf{y}) + \mathsf{F}(\mathbf{x} \vee \mathbf{y}) = \mathsf{F}(\mathbf{x}) + \mathsf{F}(\mathbf{y}). \tag{1.18}$$

Due to the fact that $\mathbf{x} \wedge \mathbf{y} + \mathbf{x} \vee \mathbf{y} = \mathbf{x} + \mathbf{y}$, each additive function is necessarily also modular.

Definition 1.60. A fusion function F is said to be *maxitive*, whenever for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$:

$$\mathsf{F}(\mathbf{x} \vee \mathbf{y}) = \mathsf{F}(\mathbf{x}) \vee \mathsf{F}(\mathbf{y}). \tag{1.19}$$

Definition 1.61. A fusion function F is said to be *minitive*, whenever for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$:

$$\mathsf{F}(\mathbf{x} \wedge \mathbf{y}) = \mathsf{F}(\mathbf{x}) \wedge \mathsf{F}(\mathbf{y}). \tag{1.20}$$

1.2.8 Other types of monotonicity

Over history, there have been different approaches to define the concept of a mean. In the Pitman sense (see Remark 1.56), a mean is meant to be translation and scale equivariant, in the Cauchy or Gini sense (see Remark 1.28) it is just an internal fusion function. Classical aggregation theory focuses on fusion functions that are monotone with respect to all their arguments. However, it is known that some classes of broadly conceived means are nonmonotone. One example of such a fusion function is the *mode*, defined as an observation that occurs most often in a data sample (in the case of unimodal data sets), see Remark 1.30. Some other examples of nonmonotone fusion functions may be found in the class of *Bajraktarević means* (see [87] and Equation (1.25)) or density-based fusion functions (see [13] as well as [51]).

As Beliakov, Calvo, and Wilkin in [43] note, unexceptional monotonicity with respect to $\leq_n$ also might not be desirable in certain contexts. For example, it can reduce the robustness of an averaging method in the case of outliers

(compare Remark 1.29). Moreover, as we shall see in further chapters, there are indeed many issues in regard to defining order preserving transformations in more complex domains than $\mathbb{I}^n$.

Due to the fact that a kind of monotonicity in the $\mathbb{I}^n$ space is nevertheless very appealing, quite recently, some researchers in aggregation theory introduced mappings that preserve orders other than $\leq_n$. Therefore, in this section we review a few of them.

The concept of weak monotonicity has been introduced by Wilkin and Beliakov in [468], see also [470]. It requires that the output of an aggregation function surely does not decrease whenever we increase all the input values by the same amount.

Definition 1.62. A fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is *weakly monotone* whenever $\mathsf{F}(\mathbf{x}+t) \geq \mathsf{F}(\mathbf{x})$ for any $t \geq 0$ and $\mathbf{x} \in \mathbb{I}^n$ such that $\mathbf{x}+t \in \mathbb{I}^n$.

Of course, each fusion function that is nondecreasing, is also weakly monotone. The same is true for any translation equivariant mapping.

In [43] it is noted that the standard nondecreasingness and weak monotonicity are two extremes of a more general situation called *monotonicity with respect to coalitions* or *quantiles* (α-monotonicity).

Definition 1.63. A fusion function $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is *monotone with respect to the α-quantile of the inputs*, $\alpha \in [0,1[$, whenever $\mathsf{F}(\mathbf{x}+t\mathbf{u}) \geq \mathsf{F}(\mathbf{x})$ for any $t \geq 0$, $\mathbf{u} \in \{0,1\}^n$ such that $\{i : u_i = 1\} \geq \lfloor \alpha n + 1 \rfloor$, and $\mathbf{x} \in \mathbb{I}^n$ such that $\mathbf{x}+t\mathbf{u} \in \mathbb{I}^n$.

What is more, Bustince, Fernandez, Kolesárová, and Mesiar introduced in [93, 94] another concept – directional monotonicity.

Definition 1.64. For a given n-dimensional vector $\vec{r} \neq \mathbf{0}$ a fusion function F is called $\vec{r}$-nondecreasing, whenever for all $t > 0$ such that $\mathbf{x} + t\vec{r} \in \mathbb{I}^n$ it holds:

$$\mathsf{F}(\mathbf{x}) \leq \mathsf{F}(\mathbf{x} + t\vec{r}). \tag{1.21}$$

Clearly, $(n*1)$-nondecreasing fusion functions are weakly monotone and vice versa. This concept is interesting if one wants to study in which *directions* a function is monotone: please notice that $\leq_n$-monotonic fusion functions are also $\vec{r}$-nondecreasing for all $\vec{r} \geq_n \mathbf{0}$. Lucca et al. in [330] called a fusion function F a *pre-aggregation mapping*, whenever it is $\vec{r}$-nondecreasing for some $\vec{r}$ and endpoint-preserving.

1.3 Construction methods

Let us discuss a few notable fusion function construction methods. Firstly, we focus on functions that are created by a fusion (composition) or modification

of other, perhaps simpler mappings. Due to that we may try to obtain data aggregation tools that start to fulfill originally missing properties or behavior.

Further on we note that many interesting fusion functions are related to universal integrals with respect to monotone measures, tools known from – among others – decision making. An appropriate choice of a monotone measure and/or integral provides us with new ways to aggregate data.

Finally, we study the concept of fusion functions which can be expressed as minimizers of some penalty.

1.3.1 Compositions and transforms of fusion functions

New fusion functions may be obtained by a proper composition of simpler ones. It turns out that under certain circumstances some of the properties of the underlying mappings may be preserved.

Proposition 1.65. *Let* $\mathsf{F} : \mathbb{I}^k \to \mathbb{I}$, $\mathsf{G}_1, \ldots, \mathsf{G}_k : \mathbb{I}^n \to \mathbb{I}$, *and* $\mathsf{H} : \mathbb{I}^n \to \mathbb{I}$ *be given by* $\mathsf{H}(\mathbf{x}) = \mathsf{F}(\mathsf{G}_1(\mathbf{x}), \ldots, \mathsf{G}_k(\mathbf{x}))$ *for* $\mathbf{x} \in \mathbb{I}^n$.

- *If* F *is* $\leq_k$*-nondecreasing and* $\mathsf{G}_1, \ldots, \mathsf{G}_k$ *are* $\leq_n$*-nondecreasing (respectively, idempotent, internal, translation equivariant, scale equivariant), then* H *is* $\leq_n$*-nondecreasing (respectively, idempotent, internal, and so forth).*
- *If* F *is* $\leq_k$*-nondecreasing and idempotent and* $\mathsf{G}_1, \ldots, \mathsf{G}_k$ *are* $\leq_n$*-nondecreasing and conjunctive (disjunctive) then* H *is also* $\leq_n$*-nondecreasing and conjunctive (respectively, disjunctive).*
- *If* F *is* $\leq_k$*-nondecreasing and* $\mathsf{G}_1, \ldots, \mathsf{G}_k$ *are* $\vec{r}$*-nondecreasing, then* H *is* $\vec{r}$*-nondecreasing, [94].*
- *If* F *is weakly monotone and* $\mathsf{G}_1, \ldots, \mathsf{G}_k$ *are translation equivariant, then* H *is weakly monotone, [468].*

Example 1.66. The Median function for even n is defined as an arithmetic mean (nondecreasing, idempotent, internal, translation, and scale equivariant) of two order statistics (which also fulfill these properties).

In particular, in Section 1.3.1.D we study an exemplary *hierarchy* of fusion functions, which leads us to the concept of an artificial neural network.

In certain contexts, it may be desirable to apply a fusion function on transformed inputs or to remap the produced outputs. For instance, we may note that nondecreasingness is a very mild condition. Because of this, we have what follows.

Proposition 1.67. *If $\varphi : \mathbb{I} \to \mathbb{I}$ is a nondecreasing univariate function with $\varphi(a) = a$ and $\varphi(b) = b$, then for each aggregation function F, $\mathsf{G} = \varphi \circ \mathsf{F}$, i.e.:*

$$\mathsf{G}(x_1, \ldots, x_n) = \varphi\left(\mathsf{F}(x_1, \ldots, x_n)\right)$$

is an aggregation function too. A similar result holds for a function given by:

$$\mathsf{H}(x_1, \ldots, x_n) = \mathsf{F}(\varphi(x_1), \ldots, \varphi(x_n)).$$

In Section 1.3.1.A we study the notion of a φ-isomorphism of a given fusion function. This shall lead us to the class of quasi-arithmetic means.

Sometimes it is also possible to transform a fusion function in such a way that its modified version starts to fulfill a desired property which was missing in the original setting.

Let $\delta_{\mathsf{F}} : \mathbb{I} \to \mathbb{I}$ denote the so-called *diagonal section* of a fusion function F, that is $\delta_{\mathsf{F}}(x) = \mathsf{F}(n{*}x)$. The following result allows us to generate an idempotent fusion function G having been given F whose diagonal section is strictly increasing and such that $\mathsf{range}(\delta_{\mathsf{F}}) = \mathsf{range}(\mathsf{F})$. Such a process is called *idempotization*.

Proposition 1.68 ([97]). *If F is such that δ_{F} is strictly increasing and there exists a fusion function G such that $\mathsf{F} = \delta_{\mathsf{F}} \circ \mathsf{G}$, then G is idempotent.*

For instance, the arithmetic mean and the geometric mean are results of idempotentization of the sum and the product, respectively.

We may also assure internality in the following way. Let F be a fusion function. Then G given for example by:

— cut-off:

$$\mathsf{G}(\mathbf{x}) = \mathsf{Min}(\mathbf{x}) \vee (\mathsf{F}(\mathbf{x}) \wedge \mathsf{Max}(\mathbf{x})),$$

or

— normalization (by, e.g., [230, Proposition 2.55]):

$$\mathsf{G}(\mathbf{x}) = \mathsf{Min}(\mathbf{x}) + (\mathsf{Max}(\mathbf{x}) - \mathsf{Min}(\mathbf{x}))\psi(\mathsf{F}(\mathbf{x})),$$

where $\psi : \mathbb{I} \to [0, 1]$ is some strictly increasing mapping, e.g., $\psi(x) = (x - a)/(b - a)$ in the case of a bounded $\mathbb{I} = [a, b]$,

is internal (recall that $\mathbb{I} = [a, b]$). Note that in both cases if F is nondecreasing, G is nondecreasing too.

Additionally, in Section 1.3.1.C we shall illustrate the concept of symmetrization.

What is more, in some applications it is useful to assume that not all the input observations have the same *impact* on the resulting value. In order to take this into account, in Section 1.3.1.B we introduce the concept of fusion functions' weighting.

Bullen [87, page 60] notes:

> *[The arithmetic mean] is the simplest mean and by far the most common; in fact for a non-mathematician this is probably the only concept for averaging a set of numbers. The arithmetic mean of two numbers a and b, $(a+b)/2$, was known and used by the Babylonians in 7000 B.C., and occurs in several contexts in the works of the Pythagorean school, sixth-fifth century B.C. [...] Aristotle, [...] used the arithmetic mean but did not give it this name. [...] The idea of arithmetic mean is also found in the concept of centroid used by Heron, and earlier by Archimedes in the third century B.C. [...]*

In the sequel we consecutively modify AMean so that we approach more and more complex (and thus interesting) fusion functions. Despite its first-glance simplicity, we shall notice that the arithmetic mean is in fact a "sleeping beauty".

A. φ-isomorphisms: Quasi-arithmetic means

Let us first introduce the notion of a φ-isomorphism.

Definition 1.69. Let $\mathbb{I} = [a, b], \mathbb{J} = [a', b']$, and $\varphi : \mathbb{I} \to \mathbb{J}$ be a strictly monotone bijection. Then the φ-isomorphism of a fusion function $\mathsf{F} : \mathbb{J}^n \to \mathbb{J}$ is a fusion function $\mathsf{F}_{[\varphi]} : \mathbb{I}^n \to \mathbb{I}$ defined as:

$$\mathsf{F}_{[\varphi]}(x_1, \ldots, x_n) = \varphi^{-1}\left(\mathsf{F}(\varphi(x_1), \ldots, \varphi(x_n))\right). \tag{1.22}$$

For instance, on $\mathbb{I} = \mathbb{J} = [a, b]$, we have $\mathsf{Max}(\mathbf{x}) = b + a - \mathsf{Min}(b + a - \mathbf{x})$. Thus, Max is a $(x \mapsto b + a - x)$-isomorphism of Min.

We have the following result, compare also Proposition 1.67.

Proposition 1.70. *If $\varphi : \mathbb{I} \to \mathbb{J}$ is a strictly monotone bijection and $\mathsf{F} : \mathbb{J}^n \to \mathbb{J}$ is an idempotent aggregation function, then $\mathsf{F}_{[\varphi]} : \mathbb{I}^n \to \mathbb{I}$ is an idempotent aggregation function too. Moreover, in the case of a weakly monotone fusion function F the same is true whenever φ is linear (but not in general), see [468].*

This serves as a basis for the definition of quasi arithmetic means, which have already been studied in the 1930s [292, 370] by, e.g., Kolmogorov and Nagumo.

Definition 1.71. Let $\varphi : \mathbb{I} \to \bar{\mathbb{R}}$ be a continuous and strictly monotonic function. Then a *quasi-arithmetic mean* generated by φ is a fusion function $\mathsf{QAMean}_\varphi : \mathbb{I}^n \to \mathbb{I}$ given by:

$$\mathsf{QAMean}_\varphi(\mathbf{x}) = \varphi^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\varphi(x_i)\right). \tag{1.23}$$

In other words, a quasi arithmetic mean is a φ-isomorphism of (the nondecreasing and idempotent) $\mathsf{AMean} : \bar{\mathbb{R}}^n \to \bar{\mathbb{R}}$. We have $\mathsf{QAMean}_\varphi = \mathsf{AMean}_{[\varphi]}$.

Table 1.1. Examples of quasi-arithmetic means under the assumption that $\mathbb{I} = [0, b]$ for some $b > 0$.

$\varphi(x)$	**name**	$\mathsf{QAMean}_\varphi(\mathbf{x})$
x	arithmetic mean	$\mathsf{AMean}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i$
x^2	quadratic mean	$\mathsf{QMean}(\mathbf{x}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}$
$1/x$	harmonic mean	$\mathsf{HMean}(\mathbf{x}) = \frac{1}{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{x_i}}$
x^r, $r \neq 0$	power mean	$\mathsf{PMean}_r(\mathbf{x}) = \left(\frac{1}{n}\sum_{i=1}^{n} x_i^r\right)^{1/r}$
$\log x$	geometric mean	$\mathsf{GMean}(\mathbf{x}) = \left(\prod_{i=1}^{n} x_i\right)^{1/n}$
$e^{\gamma x}$, $\gamma \neq 0$	exponential mean	$\mathsf{EMean}_\gamma(\mathbf{x}) = \frac{1}{\gamma}\log\left(\frac{1}{n}\sum_{i=1}^{n} e^{\gamma x_i}\right)$

Table 1.1 lists notable instances of quasi-arithmetic means. Like in [87, 230], we assume that $a = 0$, i.e., $\mathbb{I} = [0, b]$ for some $b > 0$. Note that among power means we have the arithmetic, quadratic, and harmonic means and that power means for $r \geq 1$ are actually norms.

Example 1.72. Suppose that a driver uses a cruise control device while driving a freeway. He/she always drives with the same speed at a fixed distance. Assuming that the consecutive speeds are $x_1, \ldots, x_n$, the average speed is equal to $\mathsf{HMean}(\mathbf{x})$.

Example 1.73. The exponential mean with $\gamma = 1$ (the so-called LogSumExp function) is used in certain optimization tasks (e.g., in some machine learning algorithms) as a smooth, strictly increasing, and convex approximation to the Max function. It is because for any $\mathbf{x} \in \mathbb{I}^n$ it holds $\mathsf{Max}(\mathbf{x}) \leq \mathsf{EMean}_1(\mathbf{x}) \leq \mathsf{Max}(\mathbf{x}) + \log n$.

Remark 1.74. Each quasi-arithmetic mean is, among others, nondecreasing, continuous, idempotent, and symmetric, see, e.g., [4]. Moreover, the arithmetic mean and all the exponential means are translation equivariant and the geometric mean as well as all the power means are scale equivariant, compare Theorems 1.131 and 1.132.

B. Weighting: Weighted quasi-arithmetic means

It is not unusual for the observations in an input vector to have a non-equal impact on data fusion results. For instance, in a decision making context, the opinions of some agents may be of greater importance than of the other ones, just as in Example 1.7. Also in physics, when there is a need to calculate the center of mass of a system of particles, we may need to take into account different "amounts of matter" constituting the objects of concern.

To quantify the degrees of importance of the aggregated entities, we may associate with each observation x_i its weight, w_i. Most commonly, a weighting vector, which must be of the same length as $\mathbf{x}$, is assumed to satisfy the following conditions.

Definition 1.75. We call $\mathbf{w} = (w_1, \ldots, w_n)$ a *weighting vector* if for all i it holds $w_i \geq 0$ and $\sum_{j=1}^{n} w_j = 1$.

Remark 1.76. Of course, if we are given nonnegative degrees of importance $d_1, \ldots, d_n$ that do not sum up to 1, we may always create a weighting vector as $\mathbf{w} = \mathbf{d} / \sum_{i=1}^{n} d_i$, under the assumption that $(\forall i)\ d_i = 0 \Longrightarrow w_i = 1/n$.

A weighted version of quasi-arithmetic means (also known as quasi-linear means) was introduced by Kitagawa in [274].

Definition 1.77. Let $\varphi : \mathbb{I} \to \bar{\mathbb{R}}$ be a continuous and strictly monotonic function and $\mathbf{w}$ be a weighting vector. Then a *weighted quasi-arithmetic mean* generated by φ and $\mathbf{w}$ is a fusion function $\mathsf{WQAMean}_{\varphi,\mathbf{w}} : \mathbb{I}^n \to \mathbb{I}$ given by:

$$\mathsf{WQAMean}_{\varphi,\mathbf{w}}(\mathbf{x}) = \varphi^{-1}\left(\sum_{i=1}^{n} w_i \varphi(x_i)\right) = \varphi^{-1}\left(\mathbf{w}^T \varphi(\mathbf{x})\right). \tag{1.24}$$

Clearly, if for all i it holds $w_i = 1/n$, then a weighted quasi-arithmetic mean reduces to a quasi-arithmetic mean. Among examples of such fusion functions we have, e.g.:

— $\mathsf{WAMean}_{\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i = \mathbf{w}^T \mathbf{x}$,
(weighted arithmetic mean, convex combination of inputs)

— $\mathsf{WHMean}(\mathbf{x}) = \dfrac{1}{\sum_{i=1}^{n} w_i / x_i}$, (weighted harmonic mean)

— $\mathsf{WGMean}(\mathbf{x}) = \prod_{i=1}^{n} x_i^{w_i}$, (weighted geometric mean)

and so forth. Note that $\mathsf{WQAMean}_{\varphi}$ is a φ-isomorphism of the fusion function $\mathsf{WAMean} : \bar{\mathbb{R}}^n \to \bar{\mathbb{R}}$.

Remark 1.78. If φ^{-1} is convex, then by the Jensen inequality we have that for all weighting vectors $\mathbf{w}$:

$$\mathsf{WQAMean}_{\varphi,\mathbf{w}}(\mathbf{x}) \leq \sum_{i=1}^{n} w_i x_i = \mathsf{WAMean}(\mathbf{x}).$$

Weights may also be dependent on the order of magnitude of inputs. An intuitively appealing generalization of weighted quasi-arithmetic means (and other weighted fusion functions) may be obtained by replacing a weighting vector in Equation (1.24) with a vector of *weighting functions*. This leads to the concept of *Bajraktarević means* (compare [87]):

$$\mathsf{BajMean}_{\varphi,\mathbf{w}}(\mathbf{x}) = \varphi^{-1}\left(\frac{\sum_{i=1}^{n} \mathsf{w}_i(x_i)\varphi(x_i)}{\sum_{i=1}^{n} \mathsf{w}_i(x_i)}\right), \tag{1.25}$$

where $\mathbf{w} = (\mathsf{w}_1, \ldots, \mathsf{w}_n)$ is a vector of weighting functions, $\mathsf{w}_i : \mathbb{I} \to [0, \infty[$ for all $i \in [n]$, and $\varphi : \mathbb{I} \to \bar{\mathbb{R}}$ is a strictly monotone bijection.

Remark 1.79. The case $\varphi(x) = x$ and $(\forall i \in [n])\ \mathsf{w}_i = \mathsf{w}$ for some function w generates the so-called *mixture operator*. Also note that if $\mathbf{w}_i$ are constant functions for all i, then a Bajraktarević mean reduces to a weighted arithmetic mean. Other particular cases may be formed by, e.g., setting w_i to be power functions. In such a way we get the *Gini means*:

$$\mathsf{GiniMean}^{p,q}_{\mathbf{w}}(\mathbf{x}) = \begin{cases} \left(\frac{\sum_{i=1}^{n} w_i x_i^p}{\sum_{i=1}^{n} w_i x_i^q}\right)^{1/(p-q)} & \text{if } p \neq q, \\ \left(\prod_{i=1}^{n} x_i^{w_i x_i^p}\right)^{1/\sum_{i=1}^{n} w_i x_i^p} & \text{if } p = q, \end{cases}$$

where $\mathbf{w}$ is a weighting vector and $p, q \in \mathbb{R}$. The case $p = q - 1$ generates the so-called *Lehmer means*. Note that if $q = 0$, then a Gini mean reduces to a nondecreasing power mean.

All the Bajraktarević means are of course idempotent. On the other hand, it is quite easy to find many examples of Bajraktarević means that are not nondecreasing. More generally, Beliakov, Wilkin, and Calvo in [44, 469] studied sufficient conditions for Gini means and some other Bajraktarević means to be weakly monotone.

C. Symmetrization: OWA operators

Note that if there exists $i \neq j$ such that $w_i \neq w_j$, then a weighted quasi-arithmetic mean is no longer symmetric. However, it turns out that each symmetric fusion function F may be generated by using another function G applied to an input vector's consecutive order statistics.

Proposition 1.80 ([230]). *$\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ is symmetric if and only if there exists a function $\mathsf{G} : \mathbb{I}^n \to \mathbb{I}$ such that:*

$$\mathsf{F}(x_1, \ldots, x_n) = \mathsf{G}(x_{(1)}, \ldots, x_{(n)}).$$

Technically, note that in fact the domain of G might be set to $\{\mathbf{x} \in \mathbb{I}^n : x_1 \leq \cdots \leq x_n\} \subseteq \mathbb{I}^n$ here. In other words, each fusion function may be *symmetrized* by replacing all x_i's with $x_{(i)}$'s, i.e., ith order statistics, in its definition.

For instance, a symmetrized version of a weighted arithmetic mean is called in decision making an OWA operator:

$$\mathsf{OWA}_{\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^{n} w_i x_{(i)}. \tag{1.26}$$

Its name – ordered weighted averaging – is due to Yager [478], see also [481, 482].

Example 1.81. Weighting and symmetrization naturally occurs in a case when we aggregate elements of a multiset: identical values may occur multiple times in an input data set and we do not pay attention to their order. Let us consider a multiset $\{(1,3),(2,1),(3,4)\}$ over $\mathbb{N}$, i.e., such that we have 3 ones, 1 two, and 4 threes. Then the corresponding weighting vector may be created according to the number of occurrences of elements:

value	#occurrences	weight
1	3	0.375
2	1	0.125
3	4	0.5
Σ	8	1.0

Example 1.82. Median, WinMean, and TriMean are in fact OWA operators – they are used as *robust*, i.e., less sensitive to the presence of a few outliers, estimators of an underlying probability distribution location parameters.

Example 1.83. Let us also recall the notion of a *sample quantile* of order $\alpha \in [0,1]$. Although there are many various definitions in the literature and

implementations in statistical software packages, see [255] for a review, it is generally accepted that this kind of an aggregation function is an OWA operator given by:

$$\mathsf{Q}_\alpha(\mathbf{x}) = \begin{cases} \mathsf{Min}(\mathbf{x}) & \text{for } \alpha = 0, \\ \mathsf{Median}(\mathbf{x}) & \text{for } \alpha = 0.5, \\ \mathsf{Max}(\mathbf{x}) & \text{for } \alpha = 1, \\ \gamma\, x_{(k)} + (1-\gamma)\, x_{(k+1)} & \text{otherwise,} \end{cases}$$

for some $\gamma = \gamma(\alpha, k) \in]0, 1]$ and $k \in \{\lfloor n\alpha - 1 \rfloor, \lfloor n\alpha + 1 \rfloor\}$ such that for each fixed $\mathbf{x}$ it is a nondecreasing function of α.

More precisely, Hyndman and Fan in [255] list nine quantile function types, see Table 1.2. The first three types are discontinuous functions of α. The other types (IV-IX) define continuous quantile functions. Each of the types may exhibit different properties, either algebraic or probabilistic. For instance, type VIII is approximately median-unbiased regardless of the distribution of input data (in an i.i.d. model). R by default uses type VII. Types I, III, and IV are not nondecreasing functions of α, therefore and not appropriate from our perspective.

D. Hierarchies of fusion functions

Being inspired by Torra's [443] paper, let us consider the concept of a general fusion function hierarchy, see Figure 1.1.

Definition 1.84. A hierarchy of fusion functions is a tuple $\mathcal{F} = (l, \mathbf{m}, \mathbf{F})$, where $l \in \mathbb{N}$ denotes the number of layers, $\mathbf{m} = (m_0, m_1, \ldots, m_l) \in \mathbb{N}^l$, where m_i gives the number of fusion functions in layer $i \in [l]$, $m_l = 1$ and $m_0 = n$ is the number of inputs, and $\mathbf{F} = (\mathsf{F}_j^{(i)})_{i\in[l], j\in[m_i]}$ is a sequence of fusion functions like $\mathsf{F}_j^{(i)} : \mathbb{I}^{m_{i-1}} \to \mathbb{I}$.

A hierarchy of fusion functions $\mathcal{F}$ determines in fact a new fusion function, F, whose output is determined as follows.

Algorithm 1.85. *To determine the output of a hierarchy of fusion functions* $\mathcal{F} = (l, \mathbf{m}, \mathbf{F})$*, do:*

1. *Let* $y_j^{(0)} := x_j$ *for* $j \in [n]$*;*
2. *For* $i = 1, 2, \ldots, l$ *do:*
   2.1. *For* $j = 1, 2, \ldots, m_i$ *do:*
      2.1.1. *Let* $y_j^{(i)} := \mathsf{F}_j^{(i)}(y_1^{(i-1)}, \ldots, y_{m_{i-1}}^{(i-1)})$*;*
3. *Return* $y_1^{(l)}$ *as result;*

Table 1.2. Different quantile functions listed in [255], see Example 1.83.

method	**parameters**	
I*	$k = \lfloor n\alpha \rfloor$	$\gamma = \begin{cases} 1 & \text{if } k = n\alpha \\ 0 & \text{otherwise} \end{cases}$
II	$k = \lfloor n\alpha \rfloor$	$\gamma = \begin{cases} 0.5 & \text{if } k = n\alpha \\ 0 & \text{otherwise} \end{cases}$
III*	$k = \lfloor n\alpha - 0.5 \rfloor$	$\gamma = \begin{cases} 1 & \text{if } k = n\alpha - 0.5 \text{ and } k \text{ is even} \\ 0 & \text{otherwise} \end{cases}$
IV*	$k = \lfloor n\alpha \rfloor$	$\gamma = k + 1 - \alpha n$
V	$k = \lfloor n\alpha + 0.5 \rfloor$	$\gamma = k + 1 - \alpha n - 0.5$
VI	$k = \lfloor n\alpha + \alpha \rfloor$	$\gamma = k + 1 - \alpha n - \alpha$
VII	$k = \lfloor n\alpha + 1 - \alpha \rfloor$	$\gamma = k - \alpha n + \alpha$
VIII	$k = \lfloor n\alpha + \frac{p+1}{3} \rfloor$	$\gamma = k + 1 - \alpha n - \frac{p+1}{3}$
IX	$k = \lfloor n\alpha + \frac{p}{4} + \frac{3}{8} \rfloor$	$\gamma = k + \frac{5}{8} - \alpha n - \frac{p}{4}$

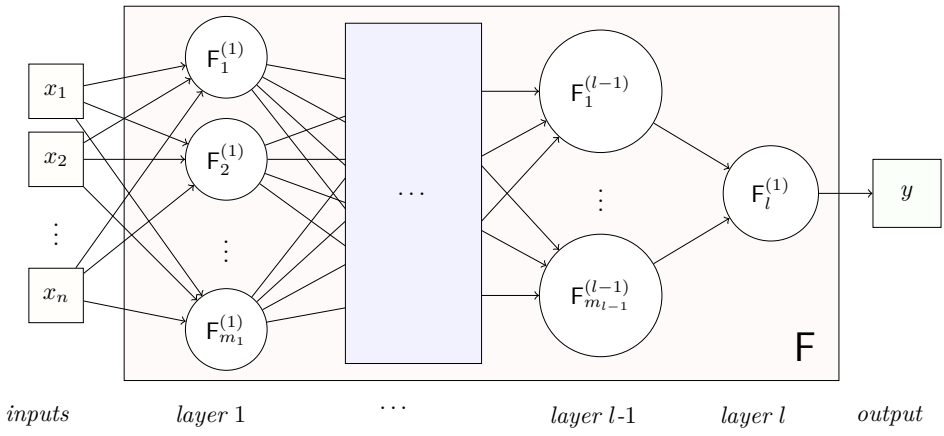


Figure 1.1. A hierarchy of fusion functions.

Surely, $l = 1$ gives the case of an ordinary, "single fusion function" setting. Please note that if $\mathsf{F}_j^{(i)}$ consequently fulfills certain properties, then by recursively applying Proposition 1.65 we may deduce the implied properties of the outcoming F.

Example 1.86. Let $\mathbb{I} = [-1, 1]$. A feedforward neural network $(l, \mathbf{m}, \mathbf{p})$, see [242], is a particular hierarchy of fusion functions $(l, \mathbf{m}, \mathbf{F})$ with:

$$\mathsf{F}_j^{(i)}(x_1, \ldots, x_{m_{i-1}}) = \mathsf{f}\left(\sum_{k=1}^{m_{i-1}} p_{k,j}^{(i)} x_k + p_{0,j}^{(i)} 1 \right),$$

where $\mathsf{f} : \mathbb{R} \to \mathbb{I}$ is the so-called activation function, typically:

$$\mathsf{f}(x) = \frac{1}{1 + \exp(-x)},$$

i.e., the sigmoidal function, and $p_{k,j}^{(i)} \in \mathbb{R}$ are arbitrary coefficients, $i \in [l]$, $j \in [m_i]$, $k \in [0 : m_{i-1}]$. Here, $y_j^{(i)}$ are called *neurons*. Note that $p_{0,j}^{(i)}$ may be treated as a coefficient standing near a so-called bias neuron, whose value is fixed at 1.

Artificial neural networks are widely used in (deep) machine learning for automated data classification.

Example 1.87. Torra in [443] showed that a feedforward neural network is isomorphic to a hierarchy of quasi-arithmetic means. Here is a sketch of its possible construction. First of all, every input element is copied with sign changed so that only nonnegative coefficients may from now on be taken into account. Then, another artificial neuron is added and the coefficients are accordingly normalized – now they are indeed weights (thus, they sum up to 1). Further on, by an appropriate choice of the generator function φ, closely related to the activation mapping, we may note that in fact only functions of quasi-arithmetic means are used.

1.3.2 Monotone measures and integrals

It turns out that some fusion functions are tightly related to monotone measures and respective integrals – tools known from decision making, social choice theory, as well as engineering. Here we shall present the notion of a universal integral, which gives a common framework to the famous Choquet [122], Sugeno [437], and Shilkret [424] integrals. Due to this, we may not only explore new interpretations of already introduced data fusion tools, but also generate new ones.

First we shall review some basic definitions and concepts, see also, e.g., [230, Chapter 5] or [39, Chapter 4]. Let $(\Omega, \mathcal{F})$ be a measurable space, i.e., a nonempty set Ω equipped with a σ-algebra.

Definition 1.88. We call $\mu : \mathcal{F} \to [0, \infty]$ a *monotone measure* (a capacity) on $(\Omega, \mathcal{F})$, if:

(a) $\mu(\emptyset) = 0$,

(b) $\mu(\Omega) > 0$, and

(c) $\mu(U) \leq \mu(V)$ for $U \subseteq V$, $U, V \in \mathcal{F}$.

Note that a monotone measure is not necessarily (σ-)additive. A normalized monotone measure, i.e., one which has $\mu(\Omega) = 1$ from now on shall be called a *fuzzy measure*.

Denoting by $\mathcal{B}([0, \infty])$ the σ-algebra of Borel subsets of $[0, \infty]$, we say that a function $\mathsf{X} : \Omega \to [0, \infty]$ is *$\mathcal{F}$-measurable*, if for each $T \in \mathcal{B}([0, \infty])$ its inverse image $\mathsf{X}^{-1}(T)$ is an element of $\mathcal{F}$.

Let $\mathcal{M}^{(\Omega,\mathcal{F})}$ denote the set of all monotone measures on $(\Omega, \mathcal{F})$ and $\mathcal{R}^{(\Omega,\mathcal{F})}$ designate the set of all $\mathcal{F}$-measurable functions $\mathsf{X} : \Omega \to [0, \infty]$.

Remark 1.89. Please note that for both $\mathcal{M}^{(\Omega,\mathcal{F})}$ and $\mathcal{R}^{(\Omega,\mathcal{F})}$ natural partial orders $\preceq_{\mathcal{M}}$ and $\preceq_{\mathcal{R}}$, respectively, may be constructed. This is because we have, e.g., $\mathsf{X} \preceq_{\mathcal{R}} \mathsf{Y}$ if and only if for all $\omega \in \Omega$ it holds $\mathsf{X}(\omega) \leq \mathsf{Y}(\omega)$. Moreover, the spaces $(\mathcal{M}^{(\Omega,\mathcal{F})}, \preceq_{\mathcal{M}})$ and $(\mathcal{R}^{(\Omega,\mathcal{F})}, \preceq_{\mathcal{R}})$ are lattices (see Section 1.7).

For further discussion we shall also need the notion of a pseudomultiplication operation.

Definition 1.90. A bivariate fusion function $\otimes : [0, \infty]^2 \to [0, \infty]$ is called a *pseudomultiplication* operation, whenever:

(a) it is nondecreasing in each variable, i.e., for $0 \leq x_1 \leq x_2$ and $0 \leq y_1 \leq y_2$, we have $x_1 \otimes y_1 \leq x_2 \otimes y_2$,

(b) it has 0 as the annihilator element, i.e., for all $x \in [0, \infty]$, $x \otimes 0 = 0 \otimes x = 0$,

(c) it has a neutral element $e > 0$, i.e., for all $x \in [0, \infty]$, $x \otimes e = e \otimes x = x$.

Note that $\otimes$ is neither necessarily associative nor commutative. Standard multiplication $\cdot$ ($e = 1$) and minimum $\wedge$ ($e = \infty$) are particular examples of pseudomultiplication operations. On the other hand, e.g., maximum $\vee$ does not annihilate at 0, thus does not fall into this class.

What is more, let $\{\omega \in \Omega : \mathsf{X}(\omega) \geq t\} \in \mathcal{F}$ be the so-called *t-level set* of X, $t \in [0, \infty]$.

Example 1.91. Let $(\Omega, \mathcal{F}) = ([n], 2^{[n]})$ and take any $\mathbf{x} \in \mathbb{I}^n$, $\mathbb{I} = [0, b]$. By setting $\mathsf{X}(i) = x_i$ we have that for any $t \geq 0$ the t-level set of X fulfills $\{\omega : \mathsf{X}(\omega) \geq t\} = \{i : x_i \geq t\}$, i.e., there is a one-to-one correspondence between $\mathbf{x}$ and X.

It is easily seen that $\{\omega : \mathsf{X}(\omega) \geq t\}_{t \in [0,\infty]}$ forms a left-continuous, nonincreasing chain (with respect to t). Thus,

$$\mathsf{S}^{(\mu,\mathsf{X})}(t) := \mu(\{\omega \in \Omega : \mathsf{X}(\omega) \geq t\}) \tag{1.27}$$

is a nonincreasing function of t.

Example 1.92. Let $(\Omega, \mathcal{F}, P)$ be a probability space, i.e., a measurable space equipped with a *probability measure* (a σ-additive fuzzy measure) P, see [61]. In this setting, Ω is called a *sample space*, any $\mathsf{X} \in \mathcal{R}^{(\Omega,\mathcal{F})}$ is named a (nonnegative real-valued) *random variable*, and $\mathsf{S}^{(\mu,\mathsf{X})}(t) = P(\{\omega \in \Omega : \mathsf{X}(\omega) \geq t\})$ is often shortened as $P(\mathsf{X} \geq t)$ and called a *survival function*. It might also be observed that a *cumulative distribution function* is tightly connected to it: we have $\mathsf{F}^{(\mu,\mathsf{X})}(t) = P(\mathsf{X} \leq t) = 1 - \mathsf{S}^{(\mu,\mathsf{X})}(t) + P(\{\omega \in \Omega : \mathsf{X}(\omega) = t\})$.

Note that in probability theory it is customary to say just "let X be a random variable with c.d.f. F" – to some degree the definitions of all the underlying objects may be inferred implicitly.

Example 1.93. In Example 1.91, if $\mu(U) = |U|$ for $U \in \mathcal{F}$ is the *counting measure*, $\mathsf{S}^{(\mu,\mathsf{X})}(t)$ gives us an appealing graphical representation of $\mathbf{x}_\sigma$ (in the form of a step function), where σ is a permutation that orders observations in $\mathbf{x}$ nonincreasingly, see Figure 1.2. Here, a choice of a different *symmetric monotone measure*, i.e., one such that $\mu(U) = \varphi(|U|)$ for some nondecreasing φ, $\varphi(0) = 0$, $\varphi(n) > 0$, corresponds to some transformation of the plot's y axis. Also, please refer, e.g., to [229] for a review of basic classes of discrete fuzzy measures.

As noted in [276], which function shall be called an integral of $\mathsf{X} \in \mathcal{R}^{(\Omega,\mathcal{F})}$ is still a disputable issue. Generally, it is agreed that an integral:

— should map the space $\mathcal{M}^{(\Omega,\mathcal{F})} \times \mathcal{R}^{(\Omega,\mathcal{F})}$ into $[0, \infty]$,

— should be at least nondecreasing with respect to each coordinate, and

— for $\mathsf{X} \equiv 0$ it should return the value 0.

In this book, we rely on the notion of a universal integral, introduced by Klement, Mesiar, and Pap. The following characterization (for the purpose of this book, we use it as a definition) was provided for it in [276, Proposition 2.7], see also [232] for an alternative setting in the discrete (thus, particular) case.

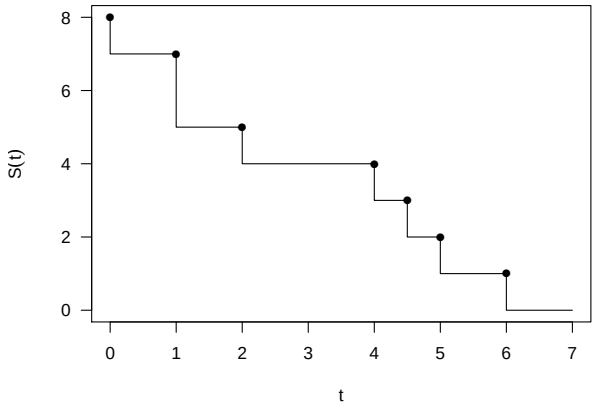


Figure 1.2. A graphical representation of an ordered version of the numeric list $(1, 6, 4.5, 1, 0, 5, 2, 4)$.

Definition 1.94. A *universal integral* corresponding to a pseudomultiplication operation $\otimes$ is a function $\mathcal{I} : \mathcal{M}^{(\Omega,\mathcal{F})} \times \mathcal{R}^{(\Omega,\mathcal{F})} \to [0, \infty]$ given by:

$$\mathcal{I}(\mu, \mathsf{X}) = \mathcal{J}\left(\mathsf{S}^{(\mu,\mathsf{X})}\right), \tag{1.28}$$

where $\mathcal{J} : \mathcal{R}^{([0,\infty],\mathcal{B}([0,\infty]))} \to [0, \infty]$ is nondecreasing and such that for each $c, d \in [0, \infty]$ we have $\mathcal{J}(d\mathbf{1}_{(0,c]}) = c \otimes d$.

Please note that $\mathcal{I}(\mu, e\mathbf{1}_U) = \mu(U)$ for all $U \in \mathcal{F}$, where e is the neutral element of $\otimes$. Given a measurable space $(\Omega, \mathcal{F})$, below are a few well-known examples of universal integrals of $\mathsf{X} \in \mathcal{R}^{(\Omega,\mathcal{F})}$ with respect to a monotone measure $\mu \in \mathcal{M}^{(\Omega,\mathcal{F})}$:

Definition 1.95. The *Choquet integral* [122] is given by:

$$\mathrm{Ch}(\mu, \mathsf{X}) = \int_{[0,\infty]} \mathsf{S}^{(\mu,\mathsf{X})}(t)\, dt. \tag{1.29}$$

Here we have $\otimes = \cdot$ (standard multiplication). Note that this integral is defined in the same way as the one by Lebesgue, but with respect to an arbitrary monotone measure. In this regard, for brevity, we often write $\int_{[0,\infty]} \mathsf{S}^{(\mu,\mathsf{X})}(t)\, dt. = \int \mathsf{X}\, d\mu$.

Example 1.96. Referring back to the setting from Example 1.92 (a probability space), the Choquet integral corresponds to the expected value of a nonnegative random variable X. This is because $\mathbb{E}\,\mathsf{X} = \int_0^\infty P(\mathsf{X} \geq t)\, dt = \int \mathsf{X}\, dP$.

Example 1.97. In example 1.91, if $\sigma \in \mathfrak{S}_{[n]}$ is an ordering permutation of $\mathbf{x}$, assuming that $x_{\sigma(0)} = 0$, it holds:

$$\begin{aligned} \mathrm{Ch}(\mu, \mathsf{X}) &= \int_{[0,\infty]} \mathsf{S}^{(\mu,\mathsf{X})}(t)\, dt = \\ &= \sum_{i=1}^{n} \big(x_{\sigma(i)} - x_{\sigma(i-1)}\big)\mu(\{\sigma(i), \ldots, \sigma(n)\}) \\ &= \sum_{i=1}^{n} x_{\sigma(i)}\big(\mu(\{\sigma(i), \ldots, \sigma(n)\}) - \mu(\{\sigma(i+1), \ldots, \sigma(n)\})\big). \end{aligned}$$

Thus, if μ is a symmetric fuzzy measure, then the Choquet integral corresponds to some OWA operator – here a monotone measure in fact generates a weighting vector. Moreover, if μ is an additive fuzzy measure, then we get the case of a weighted arithmetic mean, WAMean, see, e.g., [334].

Definition 1.98. The *Shilkret integral* [424] is given by:

$$\mathrm{Sh}(\mu, \mathsf{X}) = \sup_{t \in [0,\infty]} \{t \cdot \mathsf{S}^{(\mu,\mathsf{X})}(t)\}, \tag{1.30}$$

with convention $0 \cdot \infty = 0$.

In this case we have $\otimes = \cdot$ as well. Following is a very intuitive example of the usefulness of the introduced concepts which we attribute to Mesiar [360].

Example 1.99 ([360]). Suppose that $\Omega = \{a, b, c\}$ represents the set of three blue-collar workers and $\mathcal{F} = 2^{\Omega}$. Let $\mu : \mathcal{F} \to [0, \infty]$ give their per-hour overall performance when they work either alone or in teams.

$U \in \mathcal{F}$	$\emptyset$	$\{a\}$	$\{b\}$	$\{c\}$	$\{a, b\}$	$\{a, c\}$	$\{b, c\}$	$\{a, b, c\}$
$\mu(U)$	0	2	3	4	7	4	5	8

We see that, due to various reasons, working together on the same task does not necessarily increase their performance additively. Hence, μ is not a measure in the classical sense.

Now let $\mathsf{X} : \Omega \to [0, \infty]$ be a function denoting each worker's availability – how many hours they can work in a certain day:

$\omega \in \Omega$	a	b	c
$\mathsf{X}(\omega)$	5	4	3

The corresponding t-level sets and $\mathsf{X}^{(\mu,\mathsf{X})}(t)$ are as follows:

T	$[0, 3]$	$]3, 4]$	$]4, 5]$	$]5, \infty]$
$\{\omega \in \Omega : \mathsf{X}(\omega) \geq t\}$, $t \in T$	$\{a, b, c\}$	$\{a, b\}$	$\{a\}$	$\emptyset$
$\mathsf{S}^{(\mu,\mathsf{X})}(t)$, $t \in T$	8	7	2	0

For instance, only a and b may work for no less than 3.5 hours that day.

Here the Shilkret integral yields the result equal to $4 \cdot 7 = 28$ – this is the best total performance under the constraint that only one group may work this day. On the other hand, the Choquet integral gives $3 \cdot 8 + 1 \cdot 7 + 1 \cdot 2 = 33$ – all the workers start their work at the beginning of the time period, and then once one of them stops, he/she goes home and does not continue that day.

Example 1.100. In Example 1.93 the two integrals have an appealing graphical interpretation: the Choquet integral corresponds to the area below the step function representing a vector $\mathbf{x}$, and the Shilkret integral is the area of the largest rectangle that can be fitted under such a function.

Let us consider an example of a universal integral that uses a different pseudomultiplication operation, $\otimes = \wedge$. Hence, its value has a quite different interpretation. Even if its present form is due to Sugeno – as noted in [230] – some of its aspects were already studied by Ky Fan in the 1940s [183].

Definition 1.101. The *Sugeno integral* [437] may be expressed as:

$$\mathrm{Su}(\mu, \mathsf{X}) \quad = \quad \sup_{t \in [0,\infty]} \{t \wedge \mathsf{S}^{(\mu,\mathsf{X})}(t)\}. \tag{1.31}$$

Example 1.102. In the setting established in Examples 1.91 and 1.93 the graphical interpretation of the discrete Sugeno integral is as follows: it is the side of the largest square that can be fitted under the step function. Here, this universal integral may be expressed as:

$$\bigvee_{i=1}^{n} x_{\sigma(i)} \wedge \mu(\{\sigma(i), \ldots, \sigma(n)\}),$$

where $\sigma \in \mathfrak{S}_{[n]}$ is the ordering permutation of a given vector $\mathbf{x}$, see, e.g., [335].

Let $\mathbf{v} \in \mathbb{I}^n$ be such that $\bigvee_{i=1}^{n} v_i = b = \sup \mathbb{I}$, and $\mathcal{A} = \{A_j\}_{j \in [k]}$, $\emptyset \neq A_j \subseteq [n]$ for some k. This integral generalizes all order statistics as well as the following fusion functions:

— $\mathsf{WMax}_{\mathbf{v}}(\mathbf{x}) = \bigvee_{i=1}^{n} v_i \wedge x_i,$ (*weighted maximum*, see [170])

— $\mathsf{WMin}_{\mathbf{v}}(\mathbf{x}) = \bigwedge_{i=1}^{n} (b - v_i) \vee x_i,$ (*weighted minimum*)

— $\mathsf{OWMax}_{\mathbf{v}}(\mathbf{x}) = \bigvee_{i=1}^{n} v_i \wedge x_{(i)}$, where $v_1 \geq \cdots \geq v_n,$

(*ordered weighted maximum, see [168]*)

— $\mathsf{OWMin}_{\mathbf{v}}(\mathbf{x}) = \bigwedge_{i=1}^{n} (b - v_i) \vee x_{(i)}$, where $v_1 \leq \cdots \leq v_n$,

(*ordered weighted minimum*)

— $\mathsf{LPF}_{\mathcal{A}}(\mathbf{x}) = \bigvee_{j=1}^{k} \bigwedge_{i \in A_j} x_i$. (*lattice polynomial function*)

Note that the class of OWMax and OWMin fusion functions coincide and for each $\mathsf{LPF}_{\mathcal{A}}$ there exists $\mathcal{B} = \{B_j\}_{j \in [l]}$, $\emptyset \neq B_j \subseteq [n]$ for some l such that $\mathsf{LPF}_{\mathcal{A}}(\mathbf{x}) = \bigwedge_{j=1}^{l} \bigvee_{i \in B_j} x_i$, see [230, Proposition 5.55].

Due to the fact that this integral is defined only using $\wedge$ and $\vee$ operations, it can be applied on purely ordinal data – we will refer back to it in Section 1.7.5. As a matter of fact, the discrete Sugeno integral may also be written as, see [230, Proposition 5.63]:

$$\mathrm{Su}(\mu, \mathbf{X}) = \bigvee_{A \subseteq [n]} \left(\bigwedge_{i \in A} x_i \wedge \mu(A) \right).$$

Hence, it is a special case of the so-called *weighted lattice polynomial functions*, given by:

$$\mathsf{WLPF}_{\mathbf{v},\mathcal{A}}(\mathbf{x}) = \bigvee_{j=1}^{k} \left(\bigwedge_{i \in A_j} x_i \wedge v_j \right), \tag{1.32}$$

for some k, $\mathcal{A} = \{A_j\}_{j \in [k]}$, $\emptyset \neq A_j \subseteq [n]$, and $\mathbf{v} \in \mathbb{I}^k$.

Remark 1.103. Many interesting applications of discrete Sugeno integrals have been reported in decision making, please refer, e.g., to [448]. Moreover, it is also used in the problem of multiple significance testing in statistics, as a measure of false discovery rate, see [52] and the issue of measuring performance of scientists, see Section 5.4 and [216, 450].

Note that not all the classes of integrals known in the literature are universal integrals. For example, decomposition integrals introduced by Even and Lehrer [182], see also [364], include the non-universal Yang's PAN [486] and Lehrer's concave [312] integral as well as the discussed above Choquet and Shilkret integral.

1.3.3 Penalty-based aggregation functions

Firstly, we shall recall the notion of a metric and a pseudometric.

Definition 1.104. A *metric* on a set Z is a function $\mathfrak{d} : Z \times Z \to [0, \infty]$ such that for any $\mathbf{x}, \mathbf{y}, \mathbf{z} \in Z$:

(a) $\mathfrak{d}$ fulfills the triangle inequality $\mathfrak{d}(\mathbf{x}, \mathbf{y}) \leq \mathfrak{d}(\mathbf{x}, \mathbf{z}) + \mathfrak{d}(\mathbf{z}, \mathbf{y})$,

(b) $\mathfrak{d}$ is symmetric, i.e., $\mathfrak{d}(\mathbf{x}, \mathbf{y}) = \mathfrak{d}(\mathbf{y}, \mathbf{x})$,

(c) it holds $\mathfrak{d}(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x} = \mathbf{y}$.

Moreover, a *pseudometric* is a function $\mathfrak{d}' : Z \times Z \to [0, \infty]$ that fulfills the triangle inequality, is symmetric, and such that for $\mathbf{x} = \mathbf{y}$ we have $\mathfrak{d}'(\mathbf{x}, \mathbf{y}) = 0$.

If $\mathfrak{d}(\mathbf{x}, \mathbf{y}) = d$, then it is customary to say that "the *distance* between $\mathbf{x}$ and $\mathbf{y}$ is d".

Notably, metrics themselves may be aggregated: if $\mathsf{F} : [0, \infty]^k \to [0, \infty]$ is nondecreasing, subadditive, and such that $\mathsf{F}(k{*}0) = 0$, then given arbitrary metrics $\mathfrak{d}^{(1)}, \ldots, \mathfrak{d}^{(k)}$ we have that $\mathfrak{d}(\mathbf{x}, \mathbf{y}) = \mathsf{F}(\mathfrak{d}^{(1)}(\mathbf{x}, \mathbf{y}), \ldots, \mathfrak{d}^{(k)}(\mathbf{x}, \mathbf{y}))$ is a metric too, see, e.g., [69, 349]. Moreover, if $\|\cdot\|$ is a norm on a vector space Z, then $\mathfrak{d}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$ is a metric. In particular, for given p from now on we denote with $\mathfrak{d}_p$ the p-norm-based metric (L_p metric). Note that all L_p metrics coincide on the vector space $\mathbb{R}$, and thus on any $Z = \mathbb{I} \subseteq \mathbb{R}$: they may be expressed as $\mathfrak{d}(x, y) = |x - y|$.

The fusion functions studied in this section may be expressed as minimizers of some kind of penalty or dissimilarity measure between the observations in an input sample and the resulting value. Intuitively, this represents the idea behind widely conceived *averaging*: we seek the y that is a good "compromise", on the whole being not "far away" from the inputs.

In this section we deal with idempotent aggregation functions. To measure the overall (aggregated) dissimilarity, here we rely on the concept of a penalty function which was introduced by Yager and Rybalov in [484] and then extended in the works of Calvo and others, see, e.g., [96, 99]. The general idea behind them is well explained, e.g., in [41]: if we have equal values on input, then the output y is the same value, we have a unanimous vote. On the other hand, if some input $x_i \neq y$, then we impose a kind of "penalty" for such a disagreement. The larger the disagreement, then the more the inputs disagree with the output and the larger the penalty.

Definition 1.105 ([96]). The function $P : \mathbb{I} \times \mathbb{I}^n \to [0, \infty]$ is a *penalty function*, whenever:

(a) $P(y; \mathbf{x}) = 0$ if $\mathbf{x} = (n * y)$,

(b) for every fixed $\mathbf{x}$, the set of minimizers of $P(y; \mathbf{x})$ is either a singleton or an interval.

Definition 1.106 ([96]). Given a penalty function P, a P-based function is defined as:

$$\mathsf{F}(\mathbf{x}) = \arg\min_y P(y; \mathbf{x}) \tag{1.33}$$

if y is the unique minimizer of $P(y; \mathbf{x})$, and $y = (u + v)/2$ if the set of minimizers is an (open or closed) interval $]u, v[$.

Based on the fact that we may always take $P(y;\mathbf{x}) = (\mathsf{F}(\mathbf{x}) - y)^2$, we have the following simple yet appealing result, which states that every idempotent fusion function is a penalty-based one.

Theorem 1.107 ([96]). *Let $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ be an idempotent function. Then there exists a penalty function P such that $\mathsf{F}(\mathbf{x}) = \arg\min_y P(y;\mathbf{x})$ for all $\mathbf{x}$.*

As a particular class of penalty functions, we may consider, e.g., one that consists of mappings given by:

$$P(y;\mathbf{x}) = \sum_{i=1}^{n} w_i p(y, x_i), \tag{1.34}$$

where $\mathbf{w}$ is a weighting vector and $p : \mathbb{I} \times \mathbb{I} \to [0, \infty]$ is a dissimilarity function that fulfills:

— $p(y, x) = 0$ if and only if $x = y$,

— $p(y, x) \geq p(y, x')$ whenever $x \geq x' \geq y$ or $x \leq x' \leq y$.

Faithful penalty functions [99] are defined via $p(y, x) = K(h(y), h(x))$ where h is continuous monotone and K is convex.

Among faithful penalty-based aggregation functions we have, e.g., the weighted arithmetic mean:

$$\mathsf{WAMean}(\mathbf{x}) = \arg\min_y \sum_{i=1}^{n} w_i p(y, x_i) = \arg\min_y \sum_{i=1}^{n} w_i (x_i - y)^2$$

and median (again note that the minimizer might not be unique):

$$\mathsf{Median}(\mathbf{x}) = \arg\min_y \sum_{i=1}^{n} |x_i - y|.$$

According to [41], these two results were already known to Laplace.

On the other hand, if, e.g., $p(y, x) = (\varphi(x) - \varphi(y))^2$, then we obtain a weighted quasi-arithmetic mean with generator φ, and if we use $p(y, x_{(i)})$ instead of $p(y, x_i)$ in Equation 1.34, then we obtain a symmetric function which, unfortunately, might not always be monotonic and well-defined, see [41]. Yet, in this way it is possible to obtain, e.g., OWA operators. Other classes of (non necessarily faithful) penalty-based aggregation functions include, e.g., deviation and entropic means, see [41] and functions generated by so-called restricted [92, 95] dissimilarity functions, see also [361].

Viewing idempotent fusion functions as minimizers of some penalty function is a very inspiring concept, especially when we shall deal with aggregation of more complex objects in the following chapters. In particular, soon we are going to consider the concept of a centroid (minimizer of the sum of squared distances), 1-median (minimizer of sums of distances), and 1-center (minimizer of maximums of distances), among others.

1.4 Extended aggregation functions

Sometimes we do not know in advance the value of n (an input vector's length) or we just would like to be "prepared" to aggregate any number of observations. Here is the definition of a data fusion tool that reflects this need.

Definition 1.108. An *extended fusion function* is a mapping $\mathsf{F}^* : \mathbb{I}^* \to \mathbb{I}$.

Recall that if X is a set, then $X^* = \bigcup_{n=2}^{\infty} X^n$ designates the family of all the vectors with elements in X of length at least 2. This is because aggregation of a single value is not particularly interesting, we usually set $\mathsf{F}(x) = x$ if it is indeed necessary.

Thus, an extended fusion function may be treated as a family of $2, 3, \ldots$-ary fusion functions, each acting on a vector of fixed arity. This may be written as:

$$\mathsf{F}^* = \left(\mathsf{F}^{(2)}, \mathsf{F}^{(3)}, \mathsf{F}^{(4)}, \ldots\right),$$

where $\mathsf{F}^{(n)} = \mathsf{F}^*|_{\mathbb{I}^n}$, i.e., a projection of F^* onto $\mathbb{I}^n$. According to [98], the concept of extended aggregation functions has been introduced by Mayor and Calvo in [355].

Example 1.109. Let us go back to the definition of the arithmetic mean. Up to now, we assumed that n is fixed. Thus, formally, we have introduced:

$$\mathsf{AMean}^{(n)}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

However, this definition may naturally be extended so that input vectors of any length are accepted:

$$\mathsf{AMean}^{*}(\mathbf{x}) = \frac{1}{|\mathbf{x}|}\sum_{i=1}^{|\mathbf{x}|} x_i,$$

which we may simply write as $\mathsf{AMean}^*(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i/n$ (but now keeping in mind that we may provide a vector of any length n on input). Note that this indeed may be expressed as a family of aggregation functions,

$$\mathsf{AMean}^* = \begin{pmatrix} (x_1, x_2) & \mapsto & \frac{1}{2}(x_1 + x_2), \\ (x_1, x_2, x_3) & \mapsto & \frac{1}{3}(x_1 + x_2 + x_3), \quad \ldots \\ \ldots \end{pmatrix}.$$

1.4.1 Weighting

Now let us go back to the definition of a weighted arithmetic mean, $\mathsf{WAMean}_{\mathbf{w}}^{(n)}(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i$, where $\mathbf{w}$ is a weighting vector of length n. The question in this

very context is of course how to extend it to the domain of tuples of arbitrary length? For that we need the following definition.

Definition 1.110. A *weighting triangle* (see [101, 355]) is a sequence $\triangle = (w_{i,n} \in [0,1] : i \in [n], n \in \{2,3,\dots\})$ with $\sum_{i=1}^{n} w_{i,n} = 1$ for all $n \geq 2$.

A weighting triangle can be represented graphically as:

$$\triangle = \begin{pmatrix} & & & w_{1,2} & & w_{2,2} & & & \\ & & w_{1,3} & & w_{2,3} & & w_{3,3} & & \\ & w_{1,4} & & w_{2,4} & & w_{3,4} & & w_{4,4} & \\ \iddots & & & & \dots & & & & \ddots \end{pmatrix} \tag{1.35}$$

Based on the notion of a weighting triangle, we are now able to define, e.g., an extended weighting arithmetic mean:

$$\mathsf{WAMean}^{*}_{\triangle}(x_1,\dots,x_n) = \sum_{i=1}^{n} w_{i,n} x_i,$$

and extended OWA (see [101, 355]) operators:

$$\mathsf{OWA}^{*}_{\triangle}(x_1,\dots,x_n) = \sum_{i=1}^{n} w_{i,n} x_{(i)}.$$

Example 1.111. A weighting triangle which corresponds to the (extended) sample median generated by an OWA operator is given by:

$$\triangle = \begin{pmatrix} & & & 0.5 & & 0.5 & & & \\ & & 0 & & 1 & & 0 & & \\ & 0 & & 0.5 & & 0.5 & & 0 & \\ 0 & & 0 & & 1 & & 0 & & 0 \\ \iddots & & & & \dots & & & & \ddots \end{pmatrix}.$$

Another example is the normalized Pascal triangle with $w_{i,n} = \binom{n-1}{i-1}/2^{n-1}$, see [49]:

$$\triangle = \begin{pmatrix} & & & & 1/2 & & 1/2 & & & & \\ & & & 1/4 & & 2/4 & & 2/4 & & & \\ & & 1/8 & & 3/8 & & 3/8 & & 1/8 & & \\ & 1/16 & & 4/16 & & 6/16 & & 4/16 & & 1/16 & \\ \iddots & & & & & \dots & & & & & \ddots \end{pmatrix}.$$

Generally, there are a few possible schemes to generate weighting triangles like $\triangle = (w_{i,n} \in [0,1] : i \in [n], n \in \{2,3,\dots\})$.

— Let $\mathbf{c} = (c_1, c_2, \dots)$ with $c_i \geq 0$ for $i = 1, 2, \dots$ and $c_1 + c_2 > 0$. Then we may set (see, e.g., [258]):

$$w_{i,n} = \frac{c_i}{\sum_{j=1}^{n} c_j}.$$

— Let $\mathsf{w} : [0,1] \to [0,1]$ be a nondecreasing function with $\mathsf{w}(0) = 0$ and $\mathsf{w}(1) = 1$. In such a case we can set (see, e.g., [68, 479]):

$$w_{i,n} = \mathsf{w}\left(\frac{i}{n}\right) - \mathsf{w}\left(\frac{i-1}{n}\right).$$

More generally, triangles of coefficients like $\triangle = (w_{i,n} : i \in [n], n \in \{2, 3, \dots\})$ (with different constraints on $w_{i,n}$) may be considered when extending, e.g., WMax or OWMax operators. In a similar way we may define a triangle weighting of functions like $\triangle = (\mathsf{w}_{i,n} : \mathbb{I} \to [0, \infty[: i \in [n], n \in \{2, 3, \dots\})$ for the purpose of defining extended Bajraktarević means.

We shall refer back to these concepts when considering the so-called α- and β-orderings in Section 3.1.2 and when studying asymptotic properties of fusion functions applied on random data in Section 4.4.

1.4.2 Arity-dependent vs arity-free properties

Formally, an abstract property P of a fusion function is a kind of logical predicate: the statement "F fulfills P" might be true or false. There is a semantic equivalence (one to one correspondence) between such a predicate and the class of fusion functions that fulfill it. By defining:

$$\mathcal{P} = \{\text{fusion function } \mathsf{F} : \mathsf{F} \text{ fulfills } P\},$$

the statements "F fulfills P" (e.g., symmetry) and "$\mathsf{F} \in \mathcal{P}$" (e.g., the class of all symmetric fusion functions) coincide.

Having said that, we may introduce the following classification of extended fusion functions' properties, see [212]. A property $\mathcal{P}$ may either be:

— an *arity-free (weak) property*, if it deals only with n-ary mappings. More precisely, it is such that for all $n, m, n \neq m$ and some $\mathsf{G}^{(n)} \in \mathcal{P}|_{\mathbb{I}^n}$ it holds:

$$\left\{\mathsf{F}^*|_{\mathbb{I}^m} : \mathsf{F}^* \in \mathcal{P},\ \mathsf{F}^*|_{\mathbb{I}^n} = \mathsf{G}^{(n)}\right\} = \left\{\mathsf{F}^*|_{\mathbb{I}^m} : \mathsf{F}^* \in \mathcal{P}\right\},$$

equivalently:

$$\left(\forall \mathsf{F}^{(2)} \in \mathcal{P}|_{\mathbb{I}^2}, \mathsf{F}^{(3)} \in \mathcal{P}|_{\mathbb{I}^3}, \dots\right) \quad \left(\mathsf{F}^{(2)}, \mathsf{F}^{(3)}, \dots\right) \in \mathcal{P},$$

— or an *arity-dependent (strong) property* otherwise.

Remark 1.112. All the properties we considered up to now are arity-free. This concerns: nondecreasingness, symmetry, translation and scale equivariance, continuity, idempotence, etc.

1.4.3 Some arity-dependent properties

Let us make a review of a few interesting arity-dependent properties. We shall begin with a stronger version of idempotency.

Definition 1.113. An extended fusion function F^* is said to be *strongly idempotent*, whenever: for all $\mathbf{x} \in \bigcup_{n=1}^{\infty} \mathbb{I}^n$ and $k > 1$ it holds:

$$\mathsf{F}^*(k * \mathbf{x}) = \mathsf{F}^*(\mathbf{x}). \tag{1.36}$$

Each strongly idempotent extended fusion function is of course idempotent. Also note that Ghiselli-Ricci [222, 223] studied the concept of asymptotic idempotency.

The following property is well known from algebra, see also [230, Definition 2.63].

Definition 1.114. We say that F^* is *associative*, if and for any $\mathbf{x}, \mathbf{y} \in \bigcup_{n=1}^{\infty} \mathbb{I}^n$ it holds:

$$\mathsf{F}^*(\mathbf{x}, \mathbf{y}) = \mathsf{F}^*\big(\mathsf{F}^*(\mathbf{x}), \mathsf{F}^*(\mathbf{y})\big), \tag{1.37}$$

with assumption $\mathsf{F}^*(x) = x$.

Remark 1.115. In order to define an associative fusion function, it is sufficient only to provide a formula/algorithm that deals with an input vector of length 2. The following recursive formula may be used to compute the value of an associative function:

$$\mathsf{F}^*(x_1, \ldots, x_n) = \mathsf{F}^*(\mathsf{F}^*(x_1, \ldots, x_{n-1}), x_n).$$

In other words, to compute $\mathsf{F}^*(x_1, \ldots, x_n)$, we may use the following algorithm:

1. Let $y := x_1$;
2. For $i = 2, 3, \ldots, n$:
   2.1. Set $y := \mathsf{F}^*(y, x_i)$;
3. Return y as result;

Example 1.116. In functional programming, the above scheme is called fold, reduce, or accumulate. Among associative fusion functions we find, e.g., Prod, Min, Sum, and $\mathsf{T}_{\mathrm{Ł}}$ – it appears as a sine qua non condition in the definitions of t-norms and t-conorms in Section 1.5.2. Below we compute the two latter functions in R by using a call to the `Reduce()` built-in.

```
Reduce("+", c(1, 2, 3, 4)) # the same as sum(c(1,2,3,4))
## [1] 10
x <- c(0.6, 0.8, 0.7, 1)
Reduce(function(x, y) # Lukasiewicz t-norm
       max(0, x+y-1), x) # i.e., max(0, sum(x)-length(x)+1)
## [1] 0.1
```

Example 1.117. Apache Hadoop Streaming API (at least as far as version 2.6 is concerned) allows to run scalable Map-Reduce (see [141]) jobs with any programs acting as the mapper and/or the reducer.

By default, an input file is processed line-by-line. Mapper programs receive appropriate chunks of the input file and their aim is to convert them to key-value pairs. Such pairs should be output to `stdout` using a form like:

```
key1 \t val1
key2 \t val2
```

Then the mappers' outputs are sorted and merged so that the reducer receives a sequence of key-value pairs (on `stdin`) which are sorted with respect to keys. Thus, a Hadoop Streaming job acts like a scalable version of:

```
cat input | mapper | sort | reducer > output
```

In particular, a simple word count job (a Hadoop "hello world"-like program) may be implemented as follows.

— Mapper:

1. For each text line `l` read from `stdin`:
   1.1. Split `l` into separate words;
   1.2. For each word `w`:
      1.2.1. Write `"w \t 1 \n"` to `stdout`;

Exemplary input (`stdin`):

```
Hello world.
World, wonderful world, hello.
```

Desired output (`stdout`):

```
hello \t 1
world \t 1
world \t 1
wonderful \t 1
world \t 1
hello \t 1
```

— Reducer:

1. Count the number `c` of consecutive key-value pairs with the same key `w`;
2. Write `"w \t c \n"` to `stdout`;

Exemplary input (`stdin`):

```
<stdout of the above exemplary mapper job>
```

Desired output (`stdout`):

```
hello \t 2
wonderful \t 1
world \t 3
```

By default, the number of mappers is set to be a function of the input file's size – the mapper jobs are executed in parallel on each available cluster node. On the other hand, most often only a single reducer job is run to collect the output of all the mappers, which often creates a performance bottleneck.

However, if an aggregation function computed by the reducer is symmetric and associative, then we may set up an additional job called *combiner*, which is performed directly on the outputs generated by mappers. Its aim is to pre-aggregate chunks of data so that the single-threaded reducer has less work to do (in our word count example the combiner is exactly the same program as the reducer). It may be observed that in such a way some significant speed ups may be obtained. An exemplary work flow is graphically depicted in Figure 1.3

Remark 1.118. Assume that we have a bivariate fusion function $\mathsf{F} : X^2 \to X$, where $X = \{a_1, \ldots, a_k\}$ is a finite set. To check whether its extension is associative, we may compute a matrix which stores the results of $\mathsf{F}(a_i, a_j)$, $i \neq j$, and then apply Light's associativity test algorithm, see, e.g., [30]. Moreover, e.g., Rajagopalan and Schulman in [401] give an approximate randomized algorithm which runs in $O(k^2 \log(1/p))$ with error probability p.

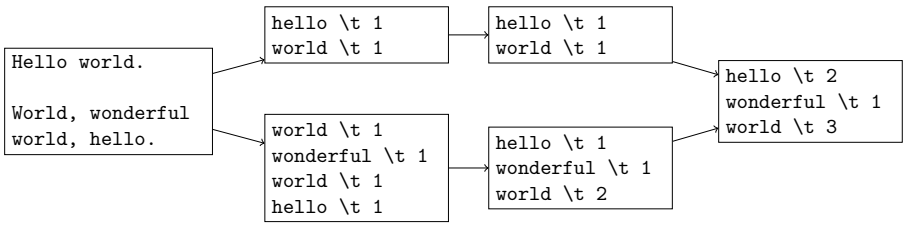


Figure 1.3. An exemplary Map-Combine-Reduce word count procedure.

A generalization of the associativity property is due to Montero and others [142] (compare also the notion of a quasi-associative fusion function – a function of k associative mappings [477]).

Definition 1.119. An extended fusion function F^* is (left)-*recursive*, whenever there exist binary functions $\psi_2, \psi_3, \cdots : \mathbb{I} \times \mathbb{I} \to \mathbb{I}$ such that for all $\mathbf{x} \in \bigcup_{n=2}^{\infty} \mathbb{I}^n$ it holds:

$$\mathsf{F}^*(x_1, \ldots, x_n) = \psi_n(\mathsf{F}^*(x_1, \ldots, x_{n-1}), x_n), \tag{1.38}$$

with assumption $\mathsf{F}^*(x) = x$.

In other words, we have:

$$\begin{aligned}
\mathsf{F}^*(x_1) &= x_1, \\
\mathsf{F}^*(x_1, x_2) &= \psi_2(\mathsf{F}^*(x_1), x_2), \\
\mathsf{F}^*(x_1, x_2, x_3) &= \psi_3(\mathsf{F}^*(x_1, x_2), x_3), \\
&\vdots \quad \vdots \quad \vdots \\
\mathsf{F}^*(x_1, x_2 \ldots, x_n) &= \psi_n(\mathsf{F}^*(x_1, x_2, \ldots, x_{n-1}), x_n).
\end{aligned}$$

Note that an associative fusion function is recursive. It is generated by setting $\psi_2 = \psi_3 = \cdots = \mathsf{F}^*|_{\mathbb{I}^2}$.

Recursive fusion functions are suitable for on-line processing of input data streams. It is only necessary to read an input stream sequentially, without the need to have it in its entirety available from the very beginning (hence, there are memory savings).

Example 1.120. The arithmetic mean is a recursive fusion function. It is because we have:

$$\mathsf{AMean}^*(x_1, \ldots, x_n) = \frac{1}{n}\Big((n-1)\mathsf{AMean}^*(x_1, \ldots, x_{n-1}) + x_n\Big).$$

Hence, in this case the generating functions are of the form:

$$\psi_n(x, y) = \frac{(n-1)x + y}{n}.$$

In a similar manner we may define a class of functions that need to have access only to the k last elements of the input sequence and/or intermediate aggregation results, for some fixed k. This leads, e.g., to the notion of aggregation of "bags" of data, as discussed by Kolesárová, Mesiar, and Montero in [291]. Here, we assume that data come in groups of a few observations.

This idea may be elaborated even further as follows. We may consider functions which require only a constant number of auxiliary variables and consecutive observations from an input data stream and thus operate in $O(1)$ memory.

Definition 1.121. We say that a function $\mathsf{F}^*: \mathbb{R}^* \to \mathbb{R}$ is *k-incremental*, $k \in \mathbb{N}$, if there exists $\mathbf{p}^{(0)} \in \mathbb{R}^k$ and a function $\psi: \mathbb{R}^k \times \mathbb{I} \to \mathbb{R}^k \times \mathbb{I}$ such that $\mathsf{F}^*(\mathbf{x})$ for any $\mathbf{x} \in \mathbb{I}^*$ may be computed as follows:

1. Let $\mathbf{p} = \mathbf{p}^{(0)} \in \mathbb{R}^k$; *(initialize auxiliary variables)*
2. For $i = 1, 2, \ldots, |\mathbf{x}|$:
   2.1. Set $(\mathbf{p}, y) := \psi(\mathbf{p}, x_i)$;
3. Return y as result;

Of course, a k-incremental fusion function is also k'-incremental for $k' > k$. Every associative fusion function is 1-incremental (store previous y), and each recursive fusion function is 2-incremental (store previous y and n).

Example 1.122. Here are some exemplary k-incremental functions (some of them are not necessarily fusion functions according to Definition 1.1):

— Sum^*, Prod^*, Max^*, Min^* are 1-incremental,

— AMean^* is 2-incremental,

— sample variance and standard deviation is 3-incremental,

— OS_k is k-incremental.

Moreover, in practice we may also be interested in fusion functions which may be computed using *online algorithms*, i.e., ones for which $\mathsf{F}^*|_{\bigcup_{i=2}^{n} \mathbb{I}^i}$ is $O(n)$-incremental. Such functions need to examine each observation only once – this is the case of the Median function, among others. The C++ Boost accumulators library includes a set of such tools.

A concept somehow related to associativity is called *decomposability* and was already studied in the 1930s by Kolmogorov [292] and Nagumo [370].

Definition 1.123. We call an extended fusion function F *decomposable* if for all $\mathbf{x} \in \mathbb{I}^*$ and $k \in [0 : |\mathbf{x}|]$ it holds:

$$\begin{aligned} & \mathsf{F}^*(x_1, \ldots, x_k, x_{k+1} \ldots, x_n) \\ = \; & \mathsf{F}^*(k * \mathsf{F}^*(x_1, \ldots, x_k), (n-k) * \mathsf{F}^*(x_{k+1}, \ldots, x_n)), \end{aligned} \tag{1.39}$$

with assumption $\mathsf{F}^*(x) = x$.

Example 1.124. It is known (see, e.g., [4]) that quasi-arithmetic means are decomposable. However, as it is noted in [230, Remark 2.70], decomposability (unlike associativity) does not determine the relationship between the result of aggregation of $n-1$ elements and n elements.

Sometimes we may also be interested in a property called (strong) *bisymmetry*, see [343].

Example 1.125 ([49]). Suppose that there are n decision makers that express their opinions on m criteria. Let x_{ij} represent the score given by the ith expert to the jth attribute. Let us suppose that we would like to compute the global score. How should we do that? There are at least three possibilities:

— Use an (nm)-ary fusion function.

— For each expert, aggregate his/her opinions on all the criteria, and then aggregate n intermediate results to obtain the global score.

— Aggregate the experts' opinions separately for each criterion. Then use another fusion function to combine such values to get a single number.

Definition 1.126. We call an extended fusion function *strongly bisymmetric*, if for all n, m, and $\mathbf{x} = (x_{i,j}) \in \mathbb{I}^{n\times m}$ it holds:

$$\begin{aligned}\mathsf{F}^*(\mathbf{x}) &= \\ &= \mathsf{F}^*\begin{pmatrix} \mathsf{F}^*(x_{1,1},\ldots,x_{1,m}), & \ldots, & \mathsf{F}^*(x_{n,1},\ldots,x_{n,m}) \end{pmatrix} \\ &= \mathsf{F}^*\begin{pmatrix} \mathsf{F}^*(x_{1,1},\ldots,x_{n,1}), & \ldots, & \mathsf{F}^*(x_{1,m},\ldots,x_{n,m}) \end{pmatrix}.\end{aligned} \tag{1.40}$$

This may be represented graphically as:

$$\begin{array}{|cccc|c|c|}\hline x_{1,1} & x_{1,2} & \ldots & x_{1,m} & \to & f_{1,\cdot} \\ x_{2,1} & x_{2,2} & \ldots & x_{2,m} & \to & f_{2,\cdot} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ x_{n,1} & x_{n,2} & \ldots & x_{n,m} & \to & f_{n,\cdot} \\ \hline \downarrow & \downarrow & & \downarrow & & \downarrow \\ \hline f_{\cdot,1} & f_{\cdot,2} & \ldots & f_{\cdot,m} & \to & f_{\cdot,\cdot} \\ \hline\end{array}$$

Moreover, we may introduce *weak bisymmetry* of an n-ary fusion function $\mathsf{F}^{(n)}$ (an arity-free property) by considering only the condition that for all $\mathbf{x} = (x_{ij}) \in \mathbb{I}^{n\times n}$ it holds:

$$\begin{aligned}&\mathsf{F}^{(n)}\begin{pmatrix} \mathsf{F}^{(n)}(x_{1,1},\ldots,x_{1,n}), & \ldots, & \mathsf{F}^{(n)}(x_{n,1},\ldots,x_{n,n}) \end{pmatrix} \\ =\ &\mathsf{F}^{(n)}\begin{pmatrix} \mathsf{F}^{(n)}(x_{1,1},\ldots,x_{n,1}), & \ldots, & \mathsf{F}^{(n)}(x_{1,n},\ldots,x_{n,n}) \end{pmatrix}.\end{aligned} \tag{1.41}$$

1.5 Choosing an aggregation method (I): Desired properties

Contrary to popular belief, aggregation is "not only about applying the arithmetic mean". We already explored (uncountably) many interesting fusion functions. One of the crucial questions is thus of course: Which one shall be chosen to reflect one's needs arising in a particular application domain?

In this section we briefly indicate a few general selection methods – each of them is based on an expected functions' behavioral specification (set up a priori). One of the possible schemes relies on known *characterization theorems*, which aim to provide a concrete definition of a class of fusion functions that fulfill a given set of properties. For instance, if we need a mapping which is symmetric, strictly increasing, continuous, idempotent, and decomposable, then by the famous Kolmogorov-Nagumo theorem (1.128) we shall conclude that we are in fact interested in one of the quasi-arithmetic means. On the other hand, if a conjunctive and – at the same time – disjunctive function is desired, then – unfortunately – it turns out that our needs are contradictory.

We shall also discuss a few notable subclasses of fusion functions, especially useful in approximate reasoning and statistics, e.g., t-norms, copulas, and fuzzy implications. These do not indicate concrete aggregation tools, but at least narrow the "search space" down.

Moreover, we sketch some numerical characteristics of fusion functions, which can also aid in the selection process. Their aim is to quantify the degree to which a function characterizes itself with a kind of behavior, what is its "typical" outcome, etc. Please keep in mind that this topic shall be significantly extended in Chapter 5.

Note that in this section we make use of a few "classical" assumptions in aggregation theory, namely that all the considered (extended) fusion functions are:

— defined for $\mathbb{I} = [0, 1]$,

— nondecreasing, and

— endpoint-preserving.

In other words, we focus on (extended) aggregation functions.

Example 1.127. Table 1.3 summarizes some of the fusion functions and their classes discussed so far along with the properties they fulfill (marked with "•"). Wherever "◕" appears in the table, there are cases in which a behavior is present as well as cases where the opposite is true. This is exactly a situation in which characterization theorems are useful.

To complete the discussion, in Section 1.6 we introduce methods for fitting fusion functions (from some classes which can be established by applying results presented in this part of the book) to empirical data.

Table 1.3. Exemplary fusion functions ($\mathbb{I} = [0, 1]$) and some basic properties they fulfill: ND – nondecreasingness, SM – symmetry, ID – idempotency, CJ – conjunctivity, DJ – disjunctivity, TE – translation equivariance, SE – scale equivariance, OE – ordinal scale equivariance.

function	ND	SM	ID	CJ	DJ	TE	SE	OE
AMean	●	●	●	○	○	●	●	○
QMean	●	●	●	○	○	○	●	○
HMean	●	●	●	○	○	○	●	○
GMean	●	●	●	○	○	○	●	○
Median	●	●	●	○	○	●	●	◕
Max	●	●	●	○	●	●	●	●
Min	●	●	●	●	○	●	●	●
$\mathsf{T}_Ł$	●	●	○	●	○	○	○	○
$\mathsf{S}_Ł$	●	●	○	○	●	○	○	○
3Π	●	●	○	○	○	○	○	○
$\mathsf{WMean}_{\mathbf{w}}$	●	◕	●	○	○	●	●	○
$\mathsf{OWA}_{\mathbf{w}}$	●	●	●	◕	◕	●	●	◕
$\mathsf{WMax}_{\mathbf{v}}$	●	◕	◕	○	◕	◕	◕	◕
$\mathsf{WMin}_{\mathbf{v}}$	●	◕	◕	◕	○	◕	◕	◕
$\mathsf{OWMax}_{\mathbf{v}}$	●	●	◕	◕	◕	◕	◕	◕
$\mathsf{QAMean}_{\varphi}$	●	●	●	○	○	◕	◕	○
$\mathsf{BajMean}_{\varphi,\mathbf{w}}$	◕	◕	●	○	○	◕	◕	○

1.5.1 Internal functions

First let us explore a few noteworthy results that concern internal (idempotent) aggregation functions. The following characterization of quasi-arithmetic means was obtained independently by Kolmogorov and Nagumo in 1930.

Theorem 1.128 ([292, 370]). *An extended fusion function* F^* *is symmetric, strictly increasing, continuous, idempotent, and decomposable if and only if there exists a continuous strictly monotonic function* $\varphi : \mathbb{I} \to \mathbb{R}$ *such that* F^* *is an extended quasi-arithmetic mean generated by* φ.

Here is a theorem by Aczel in which weak bisymmetry is substituted for decomposability.

Theorem 1.129 ([4]). *An* n*-ary fusion function* $\mathsf{F}^{(n)}$ *is strictly increasing, continuous, idempotent, and weakly bisymmetric if and only if there exists a continuous strictly monotonic function* $\varphi : \mathbb{I} \to \mathbb{R}$ *and a weighting vector* $\mathbf{w} > 0$ *such that* $\mathsf{F}^{(n)}$ *is a weighted quasi-arithmetic mean generated by* φ *and* $\mathbf{w}$.

Corollary 1.130 ([4]). *An n-ary fusion function $\mathsf{F}^{(n)}$ is symmetric, strictly increasing, continuous, idempotent, and weakly bisymmetric if and only if there exists a continuous strictly monotonic function $\varphi : \mathbb{I} \to \mathbb{R}$ such that $\mathsf{F}^{(n)}$ is an quasi-arithmetic mean generated by φ.*

According to [230], here is how Nagumo [370] characterized all the quasi-arithmetic means that fulfill translation and scale equivariance.

Theorem 1.131. *A n-ary quasi-arithmetic mean $\mathsf{QAMean}^{(n)}$ is translation equivariant if and only if it is either the arithmetic mean or it is an exponential mean.*

Theorem 1.132. *A n-ary quasi-arithmetic mean $\mathsf{QAMean}^{(n)}$ is scale equivariant if and only if it is either the geometric mean or it is a power mean.*

Recall that among power means we have the arithmetic, quadratic, and harmonic means. Taking the two above results into account we imply that the only quasi-arithmetic mean that is interval scale equivariant is the arithmetic mean.

Let us now discuss additivity and related concepts.

Theorem 1.133. *An n-ary fusion function $\mathsf{F}^{(n)}$ is additive, nondecreasing, and idempotent if and only if $\mathsf{F}^{(n)}$ is a weighted arithmetic mean.*

See [230, Proposition 4.21] for a proof. As a corollary, we have that an n-ary fusion function $\mathsf{F}^{(n)}$ is additive, nondecreasing, idempotent, and symmetric if and only if it is the arithmetic mean. Moreover, please note that nondecreasingness can be replaced with continuity in this theorem.

Theorem 1.134 ([362]). *An n-ary fusion function $\mathsf{F}^{(n)}$ is modular, nondecreasing, and idempotent if and only if:*

$$\mathsf{F}^{(n)}(\mathbf{x}) = \sum_{i=1}^{n} \mathsf{f}_i(x_i)$$

for any nondecreasing $\mathsf{f}_1, \ldots, \mathsf{f}_n : \mathbb{I} \to \mathbb{I}$ such that $(\forall x \in \mathbb{I})$ $\sum_{i=1}^{n} \mathsf{f}_i(x) = x$.

Let us consider two variants of additivity. The first one assumes that the vectors on which the addition operation is applied on comonotonic vectors (see Definition 1.41).

Definition 1.135. An n-ary fusion function $\mathsf{F}^{(n)}$ is said to be *comonotonic additive*, whenever:

$$\mathsf{F}^{(n)}(\mathbf{x}+\mathbf{y}) = \mathsf{F}^{(n)}(\mathbf{x}) + \mathsf{F}^{(n)}(\mathbf{y}),$$

for all comonotonic $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$ such that $\mathbf{x}+\mathbf{y} \in \mathbb{I}^n$.

Theorem 1.136. *An n-ary fusion function $\mathsf{F}^{(n)}$ is comonotonic additive, nondecreasing, and idempotent if and only if $\mathsf{F}^{(n)}$ is a discrete Choquet integral with respect to a fuzzy measure.*

We may also deal with a symmetrized version of the additivity property.

Definition 1.137. An n-ary fusion function $\mathsf{F}^{(n)}$ is said to be *symmetric additive*, whenever:

$$\mathsf{F}^{(n)}(\mathbf{x} \overset{S}{+} \mathbf{y}) = \mathsf{F}^{(n)}(\mathbf{x}) + \mathsf{F}^{(n)}(\mathbf{y}),$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{I}^n$ such that $\mathbf{x}+\mathbf{y} \in \mathbb{I}^n$, where $\mathbf{x} \overset{S}{+} \mathbf{y} = (x_{(1)}+y_{(1)}, \ldots, x_{(n)}+y_{(n)})$.

Clearly, each symmetric additive fusion function is necessarily symmetric.

Theorem 1.138. *An n-ary fusion function $\mathsf{F}^{(n)}$ is symmetric additive, nondecreasing, and idempotent if and only if $\mathsf{F}^{(n)}$ is an OWA operator.*

For a different characterization of OWA operators, see, e.g., [196]. Let us now present a characterization concerning associativity.

Theorem 1.139. *[195, 336] An extended fusion function F^* is nondecreasing, continuous, idempotent, and associative if and only if there exist $\alpha, \beta \in \mathbb{I}$ such that:*

$$\mathsf{F}|_{\mathbb{I}^2}(x_1, x_2) = (\alpha \wedge x_1) \vee (\beta \wedge x_2) \vee (x_1 \wedge x_2).$$

Together with symmetry the above result restricts itself to the so-called α-*median*, $\mathsf{F}|_{\mathbb{I}^2}(x_1, x_2) = \mathsf{Median}(x_1, \alpha, x_2)$ for some $\alpha \in \mathbb{I}$, see [166]. Moreover, Czogała and Drewniak in [133] presented one of the possible characterizations of the Min and Max functions.

1.5.2 Conjunctive and disjunctive functions

Another set of tools in which aggregation theory is interested in consists of fuzzy logic connectives (useful in, e.g., approximate reasoning, preference modeling, etc.) and copulas (very important in probability and statistics, compare Remark 2.12), see [18, 281, 373]. Most of them are considered as binary operations on members of $\mathbb{I} = [0, 1]$, but they may be extended to $\mathbb{I}^*$ easily.

The two properties provided below are well-known from algebra.

Definition 1.140. We say that $\mathsf{F}^{(n)}$ has an *annihilator element* $h \in \mathbb{I}$, whenever for all $\mathbf{x} \in \mathbb{I}^n$ and $i \in [n]$ we have:

$$\mathsf{F}^{(n)}(x_1, x_2, \ldots, x_{i-1}, h, x_{i+1}, \ldots, x_n) = h. \tag{1.42}$$

Definition 1.141. $\mathsf{F}^{(n)}$ has a *neutral element* $e \in \mathbb{I}$, if for all $\mathbf{x} \in \mathbb{I}^n$ and $i \in [n]$ it holds:

$$\mathsf{F}^{(n)}(e, e, \ldots, e, x_i, e, \ldots, e) = x_i. \tag{1.43}$$

This property may be extended as follows.

Definition 1.142. Given an extended fusion function F^*, we call $e \in \mathbb{I}$ its *strong neutral element*, whenever for all n and $\mathbf{x}$ it holds:

$$\mathsf{F}^{(n+1)}(x_1, x_2, \ldots, x_{i-1}, e, x_i, \ldots, x_n) = \mathsf{F}^{(n)}(\mathbf{x}). \tag{1.44}$$

T-norms. Triangular norms were first introduced by Schweizer and Sklar in the context of probabilistic metric spaces (see [421]) and are used, among others, for defining intersections of fuzzy sets and modeling the conjunction operation in fuzzy logic.

Definition 1.143. An aggregation function $\mathsf{T}^{(2)} : [0,1] \times [0,1] \to [0,1]$ is a *t-norm* if for all $x, y, z \in [0,1]$ it holds:

(a) if $y \leq z$, then $\mathsf{T}^{(2)}(x, y) \leq \mathsf{T}^{(2)}(x, z)$, (nondecreasingness)

(b) $\mathsf{T}^{(2)}(x, y) = \mathsf{T}^{(2)}(y, x)$, (symmetry)

(c) $\mathsf{T}^{(2)}(x, \mathsf{T}^{(2)}(y, z)) = \mathsf{T}^{(2)}(\mathsf{T}^{(2)}(x, y), z)$, (associativity)

(d) $\mathsf{T}^{(2)}(x, 1) = x$. (neutral element 1)

Thus, a t-norm is a symmetric conjunctive aggregation function on $[0,1]^2$. It is easily seen that the restriction of any t-norm to $\{0,1\}^2$ gives us the conjunction operation known from classical Boolean logic. Moreover, each t-norm has 0 as its annihilator element, i.e., $\mathsf{T}^{(2)}(x, 0) = \mathsf{T}^{(2)}(0, x) = 0$ for all x.

Table 1.4 lists some seminal t-norms. For any t-norm $\mathsf{T}^{(2)}$ and all x, y it holds $\mathsf{T}_{\mathrm{D}}^{(2)}(x, y) \leq \mathsf{T}^{(2)}(x, y) \leq \mathsf{Min}^{(2)}(x, y)$. Moreover, we have $\mathsf{T}_{\mathrm{Ł}}^{(2)}(x, y) \leq \mathsf{Prod}^{(2)}(x, y)$.

Recall that in Proposition 1.70 we stated that for every nondecreasing fusion function $\mathsf{F}^{(n)}$, its φ-isomorphism is also nondecreasing.

Proposition 1.144. *For any strictly increasing and continuous function* $\varphi : \mathbb{I} \to \mathbb{I}$*, if* $\mathsf{F}^{(n)}$ *is conjunctive, then* $\mathsf{F}^{(n)}_{[\varphi]}$ *is conjunctive too. Moreover, if* $\mathsf{F}^{(n)}$ *is a t-norm, then* $\mathsf{F}^{(n)}_{[\varphi]}$ *is also a t-norm.*

T-conorms. First of all, let us note what follows.

Proposition 1.145. *For any strictly decreasing and continuous function* $\varphi : \mathbb{I} \to \mathbb{I}$*,* $\mathsf{F}^{(n)}$ *is disjunctive if and only if* $\mathsf{F}^{(n)}_{[\varphi]}$ *is conjunctive.*

Triangular conorms generalize the notion of the classical Boolean logic alternative operator. They are defined as $(x \mapsto 1 - x)$-isomorphisms of t-norms.

Definition 1.146. A function $\mathsf{S}^{(2)} : [0,1] \times [0,1] \to [0,1]$ is a *t-conorm* if for all $x, y, z \in [0,1]$ it holds:

(a) if $y \leq z$, then $\mathsf{S}^{(2)}(x,y) \leq \mathsf{S}^{(2)}(x,z)$, (nondecreasingness)

(b) $\mathsf{S}^{(2)}(x,y) = \mathsf{S}^{(2)}(y,x)$, (symmetry)

(c) $\mathsf{S}^{(2)}(x, \mathsf{S}^{(2)}(y,z)) = \mathsf{S}^{(2)}(\mathsf{S}^{(2)}(x,y), z)$, (associativity)

(d) $\mathsf{S}^{(2)}(x,0) = x$. (neutral element 0)

It is evident that all t-conorms are disjunctive. Table 1.5 lists a few noteworthy t-conorms – all of them are dual to respective t-norms in Table 1.4. For any t-conorm $\mathsf{S}^{(2)}$ and all x, y it holds $\mathsf{Max}^{(2)}(x,y) \leq \mathsf{S}^{(2)}(x,y) \leq \mathsf{S}^{(2)}_{\mathrm{D}}(x,y)$. Moreover, we have $\mathsf{S}^{(2)}_{\mathrm{P}}(x,y) \leq \mathsf{S}^{(2)}_{\mathrm{Ł}}(x,y)$.

Please refer to the seminal monograph of Klement, Mesiar, and Pap [277] and their so-called position papers [278–280] as well as to [230, Chapter 3] for more details on t-norms and t-conorms.

Copulas. Copulas form another group of interesting and useful aggregation functions. They may be used in probability and statistics to model dependencies between random variables, see, e.g., [373] and also Remark 2.12.

For given n, each n-copula $\mathsf{C}^{(n)} : [0,1]^n \to [0,1]$ is a cumulative distribution function of an n-dimensional random variable having uniform margins. In particular, for $n = 2$ we have what follows.

Definition 1.147. A function $\mathsf{C}^{(2)} : [0,1] \times [0,1] \to [0,1]$ is a *2-copula* if for all $x, y, x', y' \in [0,1]$ it holds:

(a) if $x \leq x'$ and $y \leq y'$, then: (2-increasingness)

$$\mathsf{C}^{(2)}(x,y) + \mathsf{C}^{(2)}(x',y') - \mathsf{C}^{(2)}(x,y') - \mathsf{C}^{(2)}(x',y) \geq 0,$$

(b) $\mathsf{C}^{(2)}(x,0) = \mathsf{C}^{(2)}(0,x) = 0$, (annihilator element)

(c) $\mathsf{C}^{(2)}(x,1) = x$. (neutral element)

Note that each t-norm fulfills conditions (b) and (c). Moreover, each 2-copula is nondecreasing and 1-Lipschitz. There are 2-copulas that are not t-norms and vice versa (see [277]). However, e.g., associative copulas are exactly 1-Lipschitz t-norms.

$\mathsf{T}_{\text{Ł}}$ and Min are particular examples of such fusion functions. By the famous Fréchet-Hoeffding theorem (compare [373]), these are the smallest and the largest copulas, respectively. Hence, copulas are conjunctive.

An important class of associative copulas consists of Archimedean ones. Let $\varphi : [0,1] \to [0,\infty[$ be a continuous, convex, and decreasing function with $\varphi(1) = 0$. Then we may define:

$$\mathsf{C}^{(2)}_{\varphi}(x,y) = \varphi^{-1}\left(\varphi(x) + \varphi(y)\right), \tag{1.45}$$

where φ^{-1}, $\varphi^{-1}(y) = \inf\{x \in [0,1] : \varphi(x) \geq y\}$, denotes the pseudoinverse of φ. Table 1.6 lists a few particular subfamilies of Archimedean copulas. Note that the Gumbel copula with $\theta = 1$ is equivalent to the Prod fusion function, which models the case of independent random variables. What is more, $\mathsf{C}^{(2)}_{\text{C},-1} \equiv \mathsf{T}^{(2)}_{\text{Ł}}$.

Another noteworthy class consists of Gaussian copulas. If Φ denotes the standard normal cumulative distribution function (note that no analytical closed-form expression exists for it) and $\Phi_{\mathbf{V}}$ denotes the joint cumulative distribution function of the bivariate normal distribution with expectation $\mathbf{0}$ and covariance matrix $\mathbf{V}$, then:

$$\mathsf{C}^{(2)}_{\text{Gauss},\mathbf{V}}(x,y) = \Phi_{\mathbf{V}}\left(\Phi^{-1}(x), \Phi^{-1}(y)\right). \tag{1.46}$$

1.5.3 Mixed, non-aggregation, and other functions

In a quite similar manner to comonotonic additivity (compare Definition 1.135), we may introduce the comonotonic maxitivity (among others).

Theorem 1.148 (see [230, Theorem 5.81]). *An n-ary fusion function $\mathsf{F}^{(n)}$ is comonotonic maxitive, $\wedge$-equivariant, and such that $\mathsf{F}^{(n)}(n*1) = 1$ if and only if $\mathsf{F}^{(n)}$ is a discrete Sugeno integral with respect to a fuzzy measure.*

Please observe that a different characterization (using nondecreasingness, $\wedge$- and $\vee$-equivariance) of the discrete Sugeno integral was proposed by Marichal in [335].

On the other hand, we may also introduce symmetrized versions of modularity, maxitivity, and minitivity (compare also Definition 1.137). Each of them implies nondecreasingness and symmetry, at least in the case $\mathbb{I} = [0,1]$ (which is fixed in this section).

Table 1.4. Exemplary t-norms.

name	**definition**
minimum	$\mathsf{Min}^{(2)}(x,y) = x \wedge y$
product	$\mathsf{Prod}^{(2)}(x,y) = xy$
Łukasiewicz	$\mathsf{T}_{\mathrm{Ł}}^{(2)}(x,y) = (x+y-1) \vee 0$
drastic	$\mathsf{T}_{\mathrm{D}}^{(2)}(x,y) = \begin{cases} 0 & \text{if } x,y \in [0,1[\\ x \wedge y & \text{if } x=1 \text{ or } y=1 \end{cases}$
Fodor	$\mathsf{T}_{\mathrm{F}}^{(2)}(x,y) = \begin{cases} 0 & \text{if } x+y \leq 1 \\ x \wedge y & \text{if } x+y>1 \end{cases}$

Table 1.5. Exemplary t-conorms.

name	**definition**
maximum	$\mathsf{Max}^{(2)}(x,y) = x \vee y$
product	$\mathsf{S}_{\mathrm{P}}^{(2)}(x,y) = x+y-xy$
Łukasiewicz	$\mathsf{S}_{\mathrm{Ł}}^{(2)}(x,y) = (x+y) \wedge 1$
drastic	$\mathsf{S}_{\mathrm{D}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x,y \in]0,1] \\ x \vee y & \text{if } x=0 \text{ or } y=0 \end{cases}$
Fodor	$\mathsf{S}_{\mathrm{F}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x+y \geq 1 \\ x \vee y & \text{if } x+y<1 \end{cases}$

Table 1.6. Exemplary Archimedean 2-copulas.

name, parameter	**definition, generator**
Clayton, $\theta \geq -1, \theta \neq 0$	$\mathsf{C}_{\mathrm{C},\theta}^{(2)}(x,y) = \left((x^{-\theta}+y^{-\theta}-1) \vee 0\right)^{-1/\theta}$, $\varphi(t) = (t^{-\theta}-1)/\theta$
Gumbel, $\theta \geq 1$	$\mathsf{C}_{\mathrm{G},\theta}^{(2)}(x,y) = \exp\left(-\left((\log 1/x)^{\theta}+(\log 1/y)^{\theta}\right)^{1/\theta}\right)$, $\varphi(t) = (\log 1/t)^{\theta}$
Frank, $\theta \neq 0$	$\mathsf{C}_{\mathrm{F},\theta}^{(2)}(x,y) = -\frac{1}{\theta}\log\left(1-\frac{(1-\exp(-\theta x))(1-\exp(-\theta y))}{1-\exp(-\theta)}\right)$, $\varphi(t) = -\log\left(\frac{1-\exp(-\theta t)}{1-\exp(-\theta)}\right)$

Theorem 1.149 ([205], see also [362]). *An n-ary fusion function* $\mathsf{F}^{(n)}$ *is symmetric modular if and only if:*

$$\mathsf{F}^{(n)}(\mathbf{x}) = \sum_{i=1}^{n} \mathsf{f}_i(x_{(i)})$$

for any nondecreasing $\mathsf{f}_1, \ldots, \mathsf{f}_n : [0,1] \to [0,1]$ *such that* $(\forall x \in [0,1])$ $\sum_{i=1}^{n} \mathsf{f}_i(x) \leq 1$.

Idempotent symmetric modular aggregation functions are called *OMA operators* (ordered modular averages) in the Mesiar and Mesiarová-Zemánková paper [362].

Theorem 1.150 ([205], see also [230]). *An n-ary fusion function* $\mathsf{F}^{(n)}$ *is symmetric minitive if and only if:*

$$\mathsf{F}^{(n)}(\mathbf{x}) = \bigwedge_{i=1}^{n} \mathsf{f}_i(x_{(i)})$$

for any nondecreasing $\mathsf{f}_1, \ldots, \mathsf{f}_n : [0,1] \to [0,1]$.

A particular subclass of minitive fusion functions, so-called effort dominating operators, see [204], shall be referred to in Section 5.4.

Theorem 1.151 ([205], see also [230]). *An n-ary fusion function* $\mathsf{F}^{(n)}$ *is symmetric maxitive if and only if:*

$$\mathsf{F}^{(n)}(\mathbf{x}) = \bigvee_{i=1}^{n} \mathsf{f}_i(x_{(i)})$$

for any nondecreasing $\mathsf{f}_1, \ldots, \mathsf{f}_n : [0,1] \to [0,1]$.

The following result is due to Gagolewski [205].

Theorem 1.152 ([205]). *For an n-ary fusion function* $\mathsf{F}^{(n)}$ *the following conditions are equivalent:*

- $\mathsf{F}^{(n)}$ *is both symmetric minitive and symmetric maxitive,*
- $\mathsf{F}^{(n)}$ *is both symmetric minitive and symmetric modular,*
- $\mathsf{F}^{(n)}$ *is both symmetric modular and symmetric maxitive,*
- $\mathsf{F}^{(n)}$ *is given by:*

$$\mathsf{F}^{(n)}(\mathbf{x}) = \bigvee_{i=1}^{n} \mathsf{f}(x_{(i)}) \wedge v_i,$$

for some nondecreasing $\mathsf{f} : [0,1] \to [0,1]$ *and* $\mathbf{v} \in [0,1]^n$ *such that* $0 \leq \mathsf{f}(0) \leq v_n \leq \cdots \leq v_1 \leq 1$.

As a corollary, the only idempotent as well as symmetric modular, minitive, and maxitive fusion function is an ordered weighted maximum (OWMax) operator.

We already considered some characterizations which takes translation, scale, interval scale, $\wedge$-, or $\vee$-equivariance into account. Let us mention the remaining property of this kind.

Theorem 1.153 ([344]). *A fusion function* F *is nondecreasing and ordinal scale equivariant if and only if* F *is a lattice polynomial function.*

Under ordinal scale equivariance, nondecreasingness and continuity coincide, see, e.g., [230, Proposition 8.13]. Note that, as showed by Marichal in [337], the only symmetric lattice polynomial functions are exactly the order statistics, OS_k, $k \in [n]$.

Here is a whole family of functions which falls into the class of "mixed" type aggregation.

Uninorms. Recall that a t-norm is a symmetric and associative aggregation function with neutral element 1. A t-conorm, on the other hand, has the neutral element 0. Here is a class of fusion functions which have a neutral element, but such that it is neither equal to 0 nor to 1.

Definition 1.154. A fusion function $\mathsf{U}^{(2)} : [0,1] \times [0,1] \to [0,1]$ is a *uninorm* if for all $x, y, z \in [0,1]$ it holds:

(a) if $y \leq z$, then $\mathsf{U}^{(2)}(x,y) \leq \mathsf{U}^{(2)}(x,z)$, (nondecreasingness)

(b) $\mathsf{U}^{(2)}(x,y) = \mathsf{U}^{(2)}(y,x)$, (symmetry)

(c) $\mathsf{U}^{(2)}(x, \mathsf{U}^{(2)}(y,z)) = \mathsf{U}^{(2)}(\mathsf{U}^{(2)}(x,y), z)$, (associativity)

(d) for some $e \in]0,1[$ it holds $\mathsf{U}^{(2)}(x,e) = x$. (neutral element $e \notin \{0,1\}$)

Here is an important result on representation of uninorms, see [230, Proposition 3.95].

Proposition 1.155. *Let* $\mathsf{U}^{(2)} : [0,1] \times [0,1] \to [0,1]$ *be a uninorm with neutral element* e*. Then there exists a t-norm* $\mathsf{T}^{(2)}$*, a t-conorm* $\mathsf{S}^{(2)}$*, and a symmetric, idempotent aggregation function* $\mathsf{A}^{(2)}$ *such that for any* $\mathbf{x} \in \mathbb{I}^2$ *it holds:*

$$\mathsf{U}^{(2)}(\mathbf{x}) = \begin{cases} \mathsf{T}^{(2)}(\mathbf{x}) & \text{if } \mathbf{x} \in [0,e]^2, \\ \mathsf{S}^{(2)}(\mathbf{x}) & \text{if } \mathbf{x} \in [e,1]^2, \\ \mathsf{A}^{(2)}(\mathbf{x}) & \text{otherwise.} \end{cases}$$

Table 1.7. Exemplary fuzzy implications.

name	**definition**
minimal	$\mathsf{I}_0^{(2)}(x,y) = \begin{cases} 1 & \text{if } x=0 \text{ or } y=1 \\ 0 & \text{otherwise} \end{cases}$
maximal	$\mathsf{I}_1^{(2)}(x,y) = \begin{cases} 0 & \text{if } x=1 \text{ and } y=0 \\ 1 & \text{otherwise} \end{cases}$
Kleene-Dienes	$\mathsf{I}_{\mathrm{KD}}^{(2)}(x,y) = (1-x) \vee y$
Łukasiewicz	$\mathsf{I}_{\mathrm{Ł}}^{(2)}(x,y) = (1-x+y) \wedge 1$
Reichenbach	$\mathsf{I}_{\mathrm{RB}}^{(2)}(x,y) = 1-x+xy$
Fodor	$\mathsf{I}_{\mathrm{F}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x \le y \\ (1-x) \vee y & \text{if } x > y \end{cases}$
Goguen	$\mathsf{I}_{\mathrm{GG}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x \le y \\ y/x & \text{if } x > y \end{cases}$
Gödel	$\mathsf{I}_{\mathrm{GD}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x \le y \\ y & \text{if } x > y \end{cases}$
Rescher	$\mathsf{I}_{\mathrm{RS}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x \le y \\ 0 & \text{if } x > y \end{cases}$
Weber	$\mathsf{I}_{\mathrm{W}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x < 1 \\ y & \text{if } x = 1 \end{cases}$
Yager	$\mathsf{I}_{\mathrm{Y}}^{(2)}(x,y) = \begin{cases} 1 & \text{if } x=0 \text{ and } y=0 \\ y^x & \text{otherwise} \end{cases}$

Thus, a uninorm is neither internal, conjunctive, nor disjunctive. The 3Π function is an exemplary uninorm.

Fuzzy implications. As it was noted earlier, even if the nondecreasingness property is very influential in aggregation theory (many of the results presented so far would not be possible to obtain without such an assumption), it should not be treated dogmatically (compare the notion of weak monotonicity, among others). Here is a useful class of functions that generalizes the concept of the Boolean logic implication operator.

Definition 1.156. A function $\mathsf{I}^{(2)} : [0,1] \times [0,1] \to [0,1]$ is a *fuzzy implication* if for all $x, y, x', y' \in [0,1]$ it holds:

(a) if $x \le x'$, then $\mathsf{I}^{(2)}(x,y) \ge \mathsf{I}^{(2)}(x',y)$, (nonincreasingness w.r.t. x)

(b) if $y \le y'$, then $\mathsf{I}^{(2)}(x,y) \le \mathsf{I}^{(2)}(x,y')$, (nondecreasingness w.r.t. y)

(c) $\mathsf{I}^{(2)}(1,1) = 1$,

(d) $\mathsf{I}^{(2)}(0,0) = 1$,

(e) $\mathsf{I}^{(2)}(1,0) = 0$.

It is easily seen that $\mathsf{I}^{(2)}(x,1) = 1$ and $\mathsf{I}^{(2)}(0,y) = 1$ for all x, y. Table 1.7 lists some exemplary fuzzy implications. The reader is referred to the monograph by Baczyński and Jayaram [18] and, e.g., to [19, 402] for a comprehensive overview of this class of fusion functions as well as its relation to t-norms, t-conorms, and other aggregation tools.

1.5.4 Andness, orness, and other numerical characteristics

In Section 5.5 we shall discuss methods for "measuring" the degree to which a fusion function obeys some particular behavior. This may be used to aid in the aggregation tool selection process too.

To get a general intuition standing behind these numerical characteristics, let us at least list a few of them here.

— Let $\mathsf{F}^{(n)}$ be an averaging aggregation function on $[0,1]^n$. Its *orness* [172] is given by:

$$\operatorname{orness}(\mathsf{F}^{(n)}) = \frac{\int_{[0,1]^n} \mathsf{F}^{(n)}(\mathbf{x})\,d\mathbf{x} - \int_{[0,1]^n} \mathsf{Min}^{(n)}(\mathbf{x})\,d\mathbf{x}}{\int_{[0,1]^n} \mathsf{Max}^{(n)}(\mathbf{x})\,d\mathbf{x} - \int_{[0,1]^n} \mathsf{Min}^{(n)}(\mathbf{x})\,d\mathbf{x}}.$$

Of course, $\operatorname{orness}(\mathsf{Min}^{(n)}) = 0$ and $\operatorname{orness}(\mathsf{Max}^{(n)}) = 1$. In a dual manner, *andness* may be defined.

— The *average orness* [185] of $\mathsf{F}^{(n)}$ is given by:

$$\operatorname{aveorness}(\mathsf{F}^{(n)}) = \int_{[0,1]^n} \frac{\mathsf{F}^{(n)}(\mathbf{x}) - \mathsf{Min}^{(n)}(\mathbf{x})}{\mathsf{Max}^{(n)}(\mathbf{x}) - \mathsf{Min}^{(n)}(\mathbf{x})}\,d\mathbf{x},$$

where we assume that $0/0 = 0$.

— For the arithmetic mean, it suffices to contaminate a single point and set it to $\pm\infty$ to obtain an infinite value. Yet, it is known that, e.g., the sample median serves as a robust estimate for the center of an empirical distribution – it needs up to roughly 50% of the data to be contaminated to change its output value drastically. The so-called *breakdown value* measures a fusion function's sensitivity to the presence of outliers, compare [159].

1.6 Choosing an aggregation method (II): Fitting fusion functions to data

Let us presume that we have established our favorite class of fusion functions (e.g., by stating a desired set of properties that must be fulfilled and then by choosing it according to one of the characterization theorems from the previous section). For simplicity, first we are going to assume that a fusion function of our interest, $\mathsf{F}_{\mathbf{w}}$, is parametrized via a weighting vector (or, more generally, a vector of some parameters) $\mathbf{w}$. For instance, it may be a weighted quasi-arithmetic mean with a fixed generator function φ (further on we shall discuss methods for automated φ selection as well). Our main concern in this section is how to choose $\mathbf{w}$.

Of course, one may rely on experts' knowledge at this point. This was the case of the aggregation method used in Ski jumping competitions, see Example 1.36. However, if the experts are unavailable, another common option is based on a methodology widely used in data mining/machine learning (see, e.g., [446]). Namely, we may obtain an (empirical) data set of input points somehow and then:

— if we have access to desired output values for corresponding input cases provided, we may rely on supervised learning-like algorithms; the weight fitting methods discussed in this section assure consistency of the obtained fusion function's outputs with prototypes at hand;

— if we do not have initial preferences towards desired output data, unsupervised learning-like techniques may be used, see, e.g., [286, 287]; note that this task is much more vague than the previous one.

Note also that other approaches may be useful, for example reinforcement learning-based ones. Nevertheless, in this monograph we are interested in examining the first scenario.

More formally, we would like to fit a fusion function $\mathsf{F}_{\mathbf{w}}$ parametrized via a vector $\mathbf{w}$ to empirical data, see, e.g., [33]. We observe $m \geq n$ input vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{I}^n$ together with m desired output values $y^{(1)}, \ldots, y^{(m)} \in \mathbb{I}$. Our task is to compute the weighting vector $\mathbf{w}$ that best "fits" the given data set. Assuming that desired input and output data are represented as matrices $\mathbf{X} \in \mathbb{I}^{n \times m}, \mathbf{Y} \in \mathbb{I}^{1 \times m}$, and that $\mathsf{F}_{\mathbf{w}}(\mathbf{X}) = [\mathsf{F}_{\mathbf{w}}(\mathbf{x}^{(1)}) \ \cdots \ \mathsf{F}_{\mathbf{w}}(\mathbf{x}^{(m)})] \in \mathbb{I}^{1 \times m}$, we are faced with a constrained optimization problem:

$$\text{minimize } E\left(\mathsf{F}_{\mathbf{w}}(\mathbf{X}), \mathbf{Y}\right) \quad \text{w.r.t. } \mathbf{w}$$

subject to some conditions on $\mathbf{w}$ that guarantee monotonicity, idempotency, or any other valuable property, where $E : \mathbb{I}^m \times \mathbb{I}^m \to [0, \infty]$ is some *loss function* (typically a function of some metric) that we shall use as a goodness-of-fit measure.

Remark 1.157. If there exists a fusion function that *interpolates* a set of prototypical observations provided ($\mathsf{F}_{\mathbf{w}}(\mathbf{X}) = \mathbf{Y}$), algorithms like those in [35, 36], where very general Lipschitz aggregation functions are fit to data, may be used. In our case, we presume that there is a kind of "noise" in the data set and it may not always be possible to find a function that goes through all the observations. In other words, we are faced with a discrete approximation task.

1.6.1 Fitting weighted arithmetic means

Let us start by examining a quite simple case of weighted arithmetic means. At this point, only some simple linear algebra and mathematical programming tools are involved in the computations. As it shall turn out below, optimization problems utilizing the most common goodness of fit measures: squared Euclidean (least squares error, LSE), Manhattan (least absolute deviation, LAD), and Chebyshev (least maximal absolute deviation, LMD) metrics reduce themselves to quadratic and linear programming tasks (see, e.g., [377]).

Remark 1.158. The discussed algorithms may also be easily modified to fit OWA operators' weights (by ordering elements in $\mathbf{X}$ appropriately). Also note that fitting WAM weights to data is a more difficult problem than performing linear regression, as in our case weights must fulfill some additional constraints.

A. Least squares fitting

Most often, we would like to find the least squares error (LSE) solution to a weight fit problem:

$$\text{minimize} \sum_{j=1}^{m} \left(\sum_{i=1}^{n} w_i x_i^{(j)} - y^{(j)} \right)^2 \quad \text{w.r.t. } \mathbf{w} \tag{1.47}$$

subject to $\mathbf{w} \geq_n \mathbf{0}$ and $\mathbf{1}^T\mathbf{w} = 1$. This task is a quadratic programming (QP) problem, see, e.g., [49, Chapter 5] or [444].

Definition 1.159. A *quadratic programming problem* may be expressed as:

$$\text{minimize } 0.5\,\mathbf{v}^T\mathbf{D}\mathbf{v} + \mathbf{c}^T\mathbf{v} + c_0 \quad \text{w.r.t. } \mathbf{v} = (v_1, \ldots, v_n)$$

subject to:

$$\begin{aligned} \mathbf{A}\mathbf{v} &\begin{matrix}\geq \\ \leq\end{matrix}_n \mathbf{b}, \\ \mathbf{v} &\leq_n \mathbf{u}, \\ \mathbf{v} &\geq_n \mathbf{l}, \end{aligned}$$

where $\mathbf{D} \in \mathbb{R}^{n\times n}$ is symmetric and positive semidefinite, $\mathbf{c} \in \mathbb{R}^n$, $c_0 \in \mathbb{R}$, $\mathbf{l} \in \bar{\mathbb{R}}^n$, $\mathbf{u} \in \bar{\mathbb{R}}^n$, $\mathbf{l} \leq_n \mathbf{u}$, and $\mathbf{A} \in \mathbb{R}^{k\times n}, \mathbf{b} \in \mathbb{R}^k$ for some $k \geq 0$.

Remark 1.160. Figure A.2 and A.3 gives the source code of an R language interface to the quadratic programming solver from the open source CGAL [442] library. The implemented algorithm is based on a generalized simplex method, see also [220, 419]. This solver has a particularly good performance for tasks with a small number of variables but large number of constraints or a large number of variables and small number of constraints. Other R QP solvers (e.g., the `solve.QP()` function from the quadprog package) either assume that $\mathbf{D}$ is (strictly) positive definite or require additional commercial software installed, e.g., CPLEX, MOSEK, or LocalSolver.

The optimization problem given by Equation (1.47) may be rewritten in terms of a QP task as follows:

$$\text{minimize } 0.5\,\mathbf{w}^T\mathbf{X}\mathbf{X}^T\mathbf{w} - (\mathbf{X}\mathbf{Y}^T)^T\mathbf{w} \quad \text{w.r.t. } \mathbf{w} \tag{1.48}$$

with 1 linear equality constraint under the assumption that $\mathbf{w} \geq_n \mathbf{0}$, see Figure A.4 for an exemplary R implementation. Note that $\mathbf{X}\mathbf{X}^T$ is surely at least positive semidefinite, see also [444] for discussion on linearly dependent rows in $\mathbf{X}$.

B. Least absolute deviation fitting

Beliakov in [37] (see also [49, Chapter 5]) considered methods for fitting aggregation operators to observed input data using the least absolute deviation (LAD, i.e., L_1 metric) criterion, which is less sensitive to outliers than the least squares error. Nevertheless, we shall note that in this setting the solutions may be ambiguous and unstable.

We aim to find a weighting vector $\mathbf{w}$ that is a solution to the optimization problem:

$$\text{minimize } \sum_{j=1}^{m} \left| \sum_{i=1}^{n} w_i x_i^{(j)} - y^{(j)} \right| \quad \text{w.r.t. } \mathbf{w} \tag{1.49}$$

subject to $\mathbf{w} \geq_n \mathbf{0}$ and $\mathbf{1}^T\mathbf{w} = 1$.

It turns out that our LAD minimization task may be translated to a linear programming (LP) problem.

Definition 1.161. A *linear programming problem* may be expressed as:

$$\text{minimize } \mathbf{c}^T\mathbf{v} + c_0 \quad \text{w.r.t. } \mathbf{v} = (v_1, \ldots, v_n)$$

subject to:

$$\begin{aligned} \mathbf{A}\mathbf{v} &\begin{matrix}\geq \\ \leq\end{matrix}_n \mathbf{b}, \\ \mathbf{v} &\leq_n \mathbf{u}, \\ \mathbf{v} &\geq_n \mathbf{l}, \end{aligned}$$

where $\mathbf{c} \in \mathbb{R}^n$, $c_0 \in \mathbb{R}$, $\mathbf{l} \in \bar{\mathbb{R}}^n$, $\mathbf{u} \in \bar{\mathbb{R}}^n$, $\mathbf{l} \leq_n \mathbf{u}$, and $\mathbf{A} \in \mathbb{R}^{k\times n}, \mathbf{b} \in \mathbb{R}^k$ for some $k \geq 0$.

Remark 1.162. The simplex or interior-point methods, among others, may be used to solve LP tasks. Note that in some LP software, like lp_solve, the condition $\mathbf{v} \geq_n \mathbf{0}$ is always implicitly assumed. Interestingly, LP tasks may also be computed by using the mentioned-above CGAL library QP solver by simply assuming that $\mathbf{D} = \mathbf{0}$.

Let us introduce $2m$ auxiliary variables r_j^+, r_j^-, $j = 1, \ldots, m$, such that $r_j^+ - r_j^- = \sum_{i=1}^n w_i x_i^{(j)} - y^{(j)}$ and $r_j^+, r_j^- \geq 0$. With this, the optimization problem given by Equation (1.49) may be rewritten, see [65, Chapter 6], [76, Chapter 6], and [63, Chapter 6], as:

$$\text{minimize} \sum_{j=1}^{m} \left(r_j^+ + r_j^-\right) \quad \text{w.r.t. } \mathbf{w}, \mathbf{r}^+, \mathbf{r}^- \tag{1.50}$$

subject to:

$$\begin{aligned} \sum_{i=1}^{n} w_i x_i^{(j)} - r_j^+ + r_j^- &= y^{(j)}, \quad j = 1, \ldots, m \\ \sum_{i=1}^{n} w_i &= 1, \\ (\mathbf{w}, \mathbf{r}^+, \mathbf{r}^-) &\geq_{n+2m} \mathbf{0}. \end{aligned}$$

Figure A.5 presents an R implementation of this LP task setup, which again is based on the CGAL QP solver.

C. Least Chebyshev metric fitting

Let us now suppose that we would like to find the least maximum absolute deviation (LMD) solution to a weight fitting problem, i.e., one that minimizes the Chebyshev L_∞ metric:

$$\text{minimize} \bigvee_{j=1}^{m} \left| \sum_{i=1}^{n} w_i x_i^{(j)} - y^{(j)} \right| \quad \text{w.r.t. } \mathbf{w} \tag{1.51}$$

subject to $\mathbf{w} \geq_n \mathbf{0}$ and $\mathbf{1}^T \mathbf{w} = 1$. It turns out that, see [76, Chapter 6] or [63, Chapter 6], the Chebyshev metric minimization task may also be represented as an LP problem. Thus, by rewriting Equation (1.51), we get what follows:

$$\text{minimize } t \quad \text{w.r.t. } \mathbf{w}, t$$

with linear constraints of the form:

$$\sum_{i=1}^{n} w_i x_i^{(j)} - t \leq y^{(j)}, \quad j = 1, \ldots, m$$

$$\begin{aligned} \sum_{i=1}^{n} w_i x_i^{(j)} + t \quad &\geq \quad y^{(j)}, \quad j = 1, \ldots, m \\ \sum_{i=1}^{n} w_i \quad &= \quad 1, \\ (\mathbf{w}, t) \quad &\geq_{n+1} \quad 0. \end{aligned}$$

Figure A.6 gives an exemplary R language implementation for least Chebyshev metric fitting.

Example 1.163. Suppose that $n = 5$ and we are given $m = 9$ toy data points as follows:

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
$x_1^{(j)}$	0.12	0.48	0.65	0.07	0.37	0.22	0.29	0.57	0.84
$x_2^{(j)}$	0.73	0.41	0.45	0.79	0.92	0.23	0.90	0.40	0.57
$x_3^{(j)}$	0.43	0.84	0.70	0.96	0.81	0.86	0.72	0.53	0.42
$x_4^{(j)}$	0.52	0.75	0.48	0.40	0.62	0.28	0.80	0.92	0.79
$x_5^{(j)}$	0.69	0.70	0.24	0.22	0.92	0.34	0.15	0.50	0.50
$y^{(j)}$	0.58	0.56	0.70	0.40	0.78	0.50	0.64	0.62	0.73

$\mathbf{Y}$ was generated in such a way that firstly $\mathbf{w} = (0.33, 0.43, 0.10, 0.08, 0.06)$ was assumed and then some random white noise was added ($\sigma = 0.05$). Here are the results of applying the above-presented algorithms (weights and corresponding errors).

E	w_1	w_2	w_3	w_4	w_5	$\mathfrak{d}_1$	$\mathfrak{d}_2$	$\mathfrak{d}_\infty$
LAD	0.1131	0.3324	0.0000	0.3460	0.2085	<u>0.6764</u>	0.3618	0.2608
LSE	0.2349	0.2026	0.2235	0.2500	0.0890	0.7654	<u>0.2882</u>	0.1583
LMD	0.1747	0.0996	0.2719	0.4538	0.0000	0.9276	0.3243	<u>0.1335</u>
—	0.3300	0.4300	0.1000	0.0800	0.0600	0.8773	0.3360	0.1997

Remark 1.164. If given exemplars have different degrees of importance, weighted goodness of fit measures can straightforwardly be incorporated into the three above optimization tasks.

1.6.2 Preservation of output rankings

Beliakov et al. in [49], see also, e.g., [37], point out that sometimes a decision modeler may be interested in preserving the ranking of outputs. To do so, we find a permutation $\sigma \in \mathfrak{S}_{[m]}$ such that $y^{(\sigma(1))} \leq \cdots \leq y^{(\sigma(m))}$. With that, we

introduce additional constrains into our optimization task:

$$\mathsf{F}_{\mathbf{w}}(\mathbf{x}^{(\sigma(j+1))}) - \mathsf{F}_{\mathbf{w}}(\mathbf{x}^{(\sigma(j))}) \geq 0 \quad \text{for } j = 1, \ldots, m-1.$$

Let $\mathbf{X}^{(-k)} = (x_{i,j})_{i\in[n],j\in[m],j\neq k}$ denote the $\mathbf{X}$ matrix with the kth column omitted. If $\mathsf{F}_{\mathbf{w}}$ is again a weighted arithmetic mean, we get further linear inequalities of the form:

$$\mathbf{w}^T \left(\mathbf{X}^{(-1)} - \mathbf{X}^{(-m)} \right) \geq 0.$$

However, let us note that some input data may lead to optimization problems that are inconsistent, i.e., that have no feasible solutions. In order to overcome this limitation we may try to incorporate an additional term into our goodness of fit measure which acts as a penalty for violating the desired output ranking:

$$\text{minimize } E\left(\mathbf{w}^T\mathbf{X}, \mathbf{Y}\right) + P\left(\mathbf{w}^T(\mathbf{X}^{(-m)} - \mathbf{X}^{(-1)}) \vee 0\right) \quad \text{w.r.t. } \mathbf{w}$$

Typically, we set $P(\mathbf{z}) = p\sum_{i=1}^{m-1} z_i^2$ or $P(\mathbf{z}) = p\sum_{i=1}^{m-1} |z_i|$ for some tuning parameter $p > 0$ that must be set up empirically, e.g., by further numeric experiments. For instance, we may try to seek the smallest p such that the Kendall correlation coefficient between $\mathbf{Y}$ and $\mathbf{w}_p^T\mathbf{X}$ is as large as possible.

A. LAD fit with P being the L_1 norm

In the case that E is the L_1 metric and $P(\mathbf{z}) = p\sum_{i=1}^{m-1} |z_i|$ we get an LP problem, see [49, page 267], which is a version of Equation (1.50) with $m-1$ additional ($n + 3m - 1$ in total) variables and exactly $n + 5m - 1$ constraints:

$$\text{minimize } \sum_{j=1}^{m} \left(r_j^+ + r_j^-\right) + p\sum_{j=1}^{m-1} q_j \quad \text{w.r.t. } \mathbf{w}, \mathbf{r}^+, \mathbf{r}^-, \mathbf{q}$$

subject to:

$$\begin{aligned}
\sum_{i=1}^{n} w_i x_i^{(j)} - r_j^+ + r_j^- &= y^{(j)}, \quad j = 1, \ldots, m \\
\sum_{i=1}^{n} w_i &= 1, \\
(\mathbf{w}, \mathbf{r}^+, \mathbf{r}^-, \mathbf{q}) &\geq_{n+3m-1} \mathbf{0}, \\
\sum_{i=1}^{n} w_i \left(x_i^{(\sigma(j+1))} - x_i^{(\sigma(j))} \right) + q_j &\geq 0, \quad j = 1, \ldots, m-1
\end{aligned}$$

where σ is an ordering permutation of $(y^{(1)}, \ldots, y^{(m)})$.

B. LSE fit with P being the squared L_2 norm

It turns out (in [49] only the case of P being the L_1 norm is considered) that the case of least squared error fitting with $P(\mathbf{z}) = p\sum_{i=1}^{m-1} z_i^2$ is quite similar to the previous one. We may incorporate $m-1$ additional variables into the quadratic programming task given by Equation (1.48) and approach the following optimization problem:

$$\text{minimize } 0.5\,\mathbf{v}^T\mathbf{D}\mathbf{v} + \mathbf{c}^T\mathbf{v} \quad \text{w.r.t. } \mathbf{v} = (\mathbf{w}, \mathbf{q})$$

subject to:

$$\begin{aligned} \sum_{i=1}^{n} w_i &= 1, \\ (\mathbf{w}, \mathbf{q}) &\geq_{2n-1} \mathbf{0}, \\ \sum_{i=1}^{n} w_i \left(x_i^{(\sigma(j+1))} - x_i^{(\sigma(j))}\right) + q_j &\geq 0, \quad j = 1, \ldots, m-1 \end{aligned}$$

where:

$$\mathbf{D} = \left[\begin{array}{c|ccc} \mathbf{X}\mathbf{X}^T & \multicolumn{3}{c}{\mathbf{0}} \\ \hline & p & \cdots & 0 \\ \mathbf{0} & 0 & \ddots & 0 \\ & 0 & \cdots & p \end{array}\right], \quad \mathbf{c} = \left[\begin{array}{c} -\mathbf{X}\mathbf{Y}^T \\ \hline \mathbf{0} \end{array}\right].$$

Example 1.165. Let us go back to the data set studied in Example 1.163. Below are the results of finding the best fitting WAM weights, together with Kendall's τ correlation coefficient between $\mathbf{Y}$ and the output generated by the computed model. Parameters p were selected so that τ is maximized and then the error metric of interest is minimized.

E	$P(\mathbf{z})$	$\mathfrak{d}_1$	$\mathfrak{d}_2$	$\mathfrak{d}_\infty$	τ
LAD	0	<u>0.6764</u>	0.3618	0.2608	0.28
LSE	0	0.7654	<u>0.2882</u>	0.1583	0.56
LMD	0	0.9276	0.3243	<u>0.1335</u>	0.33
LAD	$1.2\sum_i \lvert z_i\rvert$	0.8059	0.3775	0.2575	<u>0.72</u>
LSE	$2.8\sum_i z_i^2$	0.8914	0.3339	0.2063	<u>0.72</u>

We see that we were able to match the output ranking quite well, however, at the cost of increasing the minimized goodness-of-fit measure. Also please keep in mind that there are data sets for which we cannot increase the initial τ.

1.6.3 Regularization

A well-known fact from machine learning is that even if we establish "good" weights on a given input sample, we do not necessarily obtain a model which

exhibits satisfactory behavior on other data that come from the same source. For instance, an estimated fusion function may be overfitted. For this reason, in regression analysis the concept of *parameter regularization* is sometimes used. It has a form of an additional penalty term dependent on some norm (or its function) of the vector of parameters. And so, e.g., ridge regression aims to minimize the squared prediction error plus a properly scaled, squared L_2 norm of the variables.

In our case we may consider, for some λ, an optimization task:

$$\text{minimize } E\left(\mathsf{F}_{\mathbf{w}}(\mathbf{X}), \mathbf{Y}\right) + \lambda\|\mathbf{w}\| \quad \text{w.r.t. } \mathbf{w}$$

subject to $\mathbf{1}^T\mathbf{w} = 1$ and $\mathbf{w} \geq_n \mathbf{0}$, where $\|\cdot\|$ is some norm (or its function), typically squared L_2. Note that due to the usual constraints on $\mathbf{w}$, the use of the L_1 norm (like, e.g., in Lasso regression) does not make much sense at this point.

Incorporating the penalty term $\|\cdot\|_2^2$ in optimization tasks discussed above is relatively easy, therefore it is left to the kind reader.

Remark 1.166. Regularization in the case of WAM weights estimation works quite well if n or m is relatively small. If this is not the case, we often do not observe positive effects of introducing the mentioned penalty. Unlike in regression problems, where we always presuppose that $\lambda \geq 0$, in our framework we are bounded with the constraint $\sum_i w_i = 1$ which, for large λ, tends to generate weighting vectors such that $w_i \to 1/n$. On the other hand, in the current framework the case of $\lambda < 0$ may also lead to useful outcomes. Yet, we should note that for $\lambda \to -\infty$ we observe that $w_j \to 1$ for some $j \in [n]$.

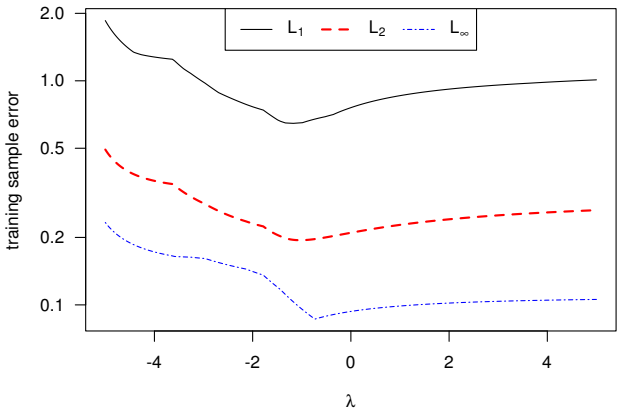


Figure 1.4. Three error measures on a test data set from Example 1.167 as a function of regularization penalty λ.

Example 1.167. Let us consider a data set generated randomly with R as follows:

```
set.seed(321)
n <- 10
m <- 100
realw <- runif(n)
realw <- realw/sum(realw)
X <- t(round(matrix(runif(n*m, 0, 1), nrow=m), 2))
Y <- t(realw) %*% X + rnorm(m, 0, 0.05)

train <- sample(1:m, m*0.8)
X_test <- X[,-train,drop=FALSE] # test sample
Y_test <- Y[,-train,drop=FALSE]
X <- X[,train,drop=FALSE]       # training sample
Y <- Y[,train,drop=FALSE]
```

The set is divided into two parts: a training sample (80% of the observations, used to compute the weights) and a test sample (20%, used to estimate the error). Here we consider a QP task:

$$\text{minimize } 0.5\,\mathbf{w}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I})\mathbf{w} - (\mathbf{X}\mathbf{Y}^T)^T\mathbf{w} \quad \text{w.r.t. } \mathbf{w} \tag{1.52}$$

subject to $\mathbf{w} \geq_n \mathbf{0}$, $\mathbf{1}^T\mathbf{w} = 1$, which optimizes the squared error plus a $\lambda\|\mathbf{w}\|_2^2$ penalty term. Figure 1.4 depicts three goodness of fit measures as a function of λ. We see that in this example we are able to improve the least squares error measure (which was minimized in this case).

E	λ	$\mathfrak{d}_1$	$\mathfrak{d}_2$	$\mathfrak{d}_\infty$
— (using `realw`)	0	0.8113	0.2256	0.1009
LAD	0	0.8525	0.2315	0.0956
LSE	0	0.7589	0.2098	<u>0.0935</u>
LMD	0	0.9137	0.2384	0.0975
LSE	−1.03	<u>0.6492</u>	<u>0.1944</u>	0.0970

1.6.4 Fitting weights of weighted quasi-arithmetic means

Let us now consider the case of $\mathsf{F}_{\mathbf{w}} = \varphi^{-1}(\mathbf{w}^T\varphi(\mathbf{x}))$, i.e., weighted quasi-arithmetic means, for an arbitrary but known and fixed continuous, strictly increasing generator function φ. Note that the case of fitting φ to empirical data is discussed later on.

Torra in [444, 445] discussed weighted quasi-arithmetic mean fitting using the L_2-metric minimization criterion. Yet, he noted that the problem is difficult in general, so he assumed that the exemplars are not subject to errors. In such a case, noting that φ is surely invertible, we have for all j:

$$\sum_{i=1}^{n} w_i\varphi(x_i^{(j)}) = \varphi(y^{(j)}).$$

Using this assumption, instead of minimizing:

$$\|\varphi^{-1}\left(\mathbf{w}^T\varphi(\mathbf{X})\right) - \mathbf{Y}\|_2$$

one can minimize a quite different (in general) goodness of fit measure:

$$\|\mathbf{w}^T\varphi(\mathbf{X}) - \varphi(\mathbf{Y})\|_2.$$

A similar approach, this time concerning the L_1 metric, was utilized by Beliakov et al. in, e.g., [37, 45, 49]. For this task, exactly the methods presented in the previous subsection can be applied, but this time on appropriately transformed $\mathbf{X}$ and $\mathbf{Y}$. Such an approach is often called *linearization* of inputs.

Let us suppose, however, that we would like to solve the original weight fit problem and not the simplified one. This leads (in general) to a nonlinear optimization task.

Example 1.168. Let $n = 5, m = 9$, and $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}$ be the same as in Example 1.163. This time, however, $\varphi(x) = x^2$ and $y_{(1)}, \ldots, y_{(m)}$ is as follows:

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
$y^{(j)}$	0.65	0.58	0.70	0.51	0.82	0.56	0.70	0.64	0.75

Here are the true $\mathfrak{d}_1$ and $\mathfrak{d}_2$ errors in the case of linearized and optimal goodness-of-fit measure minimization tasks. The differences are quite small, but not negligible. Yet, we may observe that often the linearized and "exact" E minimization tasks lead to solutions which are very close to each other.

E	$\mathfrak{d}_1$	$\mathfrak{d}_2$	$\mathfrak{d}_\infty$
LAD – linearization	0.7385	0.4120	0.2798
LSE – linearization	0.7423	0.2859	0.1626
LAD – optimal	<u>0.7157</u>	0.3170	0.2044
LSE – optimal	0.7587	<u>0.2817</u>	<u>0.1501</u>

A. LSE fit of WQAMean weights

We aim to:

$$\text{minimize} \sum_{j=1}^{m}\left(\varphi^{-1}\left(\sum_{i=1}^{n} w_i\varphi\left(x_i^{(j)}\right)\right) - y^{(j)}\right)^2 \quad \text{w.r.t. } \mathbf{w} \tag{1.53}$$

subject to $\mathbf{w} \geq_n \mathbf{0}$ and $\mathbf{1}^T\mathbf{w} = 1$. By homogeneity and triangle inequality of $\|\cdot\|_2$ we have that this is a convex optimization problem. To drop the constraints on $\mathbf{w}$, let us use an approach considered by Filev and Yager [186], see also [445] (a barrier function could also be used for that, among others). We take a different parameter space, $\boldsymbol{\lambda} \in \mathbb{R}^n$, such that:

$$w_i = \frac{\exp(\lambda_i)}{\sum_{k=1}^{n}\exp(\lambda_k)}.$$

Assuming that φ^{-1} is differentiable, let us determine the gradient $\nabla E(\boldsymbol{\lambda})$. For any $k \in [n]$ it holds:

$$\begin{aligned}\frac{\partial}{\partial \lambda_k} E(\boldsymbol{\lambda}) &= 2\frac{\exp(\lambda_k)}{\sum_{i=1}^n \exp(\lambda_i)} \sum_{j=1}^m \left(\varphi^{-1}\left(\frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_i)} \right) - y^{(j)} \right) \\ &\cdot \left(\varphi^{-1}\right)'\left(\frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_i)} \right) \\ &\cdot \left(\varphi\left(x_k^{(j)}\right) - \frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_i)} \right).\end{aligned}$$

Assuming that $\mathbf{Z} = \mathbf{w}^T\varphi(\mathbf{X})$ and $\mathbf{w} = \exp(\boldsymbol{\lambda})/\mathbf{1}^T\exp(\boldsymbol{\lambda})$, we have:

$$\nabla E(\boldsymbol{\lambda}) = 2 \cdot \mathbf{w} \cdot \left(\left(\left(\phi^{-1}(\mathbf{Z}) - Y\right) \cdot \left(\varphi^{-1}\right)'(\mathbf{Z}) \right) \times \left(\varphi(\mathbf{X})^T - \mathbf{Z} \right) \right),$$

where $\cdot, -$ stand for elementwise vectorized multiplication and subtraction, respectively, $\times$ denotes matrix multiplication, and $\varphi(\mathbf{X})^T - \mathbf{Z}$ means that we subtract $\mathbf{Z}$ from each column in $\varphi(\mathbf{X})^T$ (this is in fact how matrix and vector arithmetic operations are vectorized in R). Figure A.7 gives an R implementation of a weight fitting procedure which is based on a quasi-Newton nonlinear optimization method by Broyden, Fletcher, Goldfarb and Shanno (the BFGS algorithm, see [377]). Please note that while using the mentioned reparametrization, the BFGS algorithm may occasionally fail to converge.

B. LAD fit of WQAMean weights

Now let us:

$$\text{minimize} \sum_{j=1}^m \left| \varphi^{-1}\left(\sum_{i=1}^n w_i \varphi\left(x_i^{(j)}\right) \right) - y^{(j)} \right| \quad \text{w.r.t. } \mathbf{w} \tag{1.54}$$

subject to $\mathbf{w} \geq_n \mathbf{0}$ and $\mathbf{1}^T\mathbf{w} = 1$. This case is problematic to nonlinear solvers, as our goodness-of-fit measure is not differentiable at 0 and we observe that methods like BFGS (using numeric finite-difference approximation of the gradient) may return results that are not close enough to the optimum.

In order to overcome this limitation, we propose the following heuristic. Instead of minimizing $\sum_{j=1}^m |z_j|$, we may consider $\sum_{j=1}^m \sqrt{z_j^2 + \varepsilon^2}$ for some $\varepsilon > 0$, typically $\varepsilon = 10^{-12}$. This is because $|x| \leq \sqrt{x^2 + \varepsilon^2}$ and $\sqrt{x^2 + \varepsilon^2} \to_{\varepsilon \to 0} |x|$ for all x. Thus, our task is now to:

$$\text{minimize} \sum_{j=1}^m \sqrt{ \left(\varphi^{-1}\left(\frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_k)} \right) - y^{(j)} \right)^2 + \varepsilon^2 } \quad \text{w.r.t. } \boldsymbol{\lambda} \tag{1.55}$$

where again we use the reparametrization $w_i = \frac{\exp(\lambda_i)}{\sum_{k=1}^n \exp(\lambda_k)}$, which enables us to drop any additional constraints. In such a case we have:

$$\begin{aligned}
\frac{\partial}{\partial \lambda_k} E(\boldsymbol{\lambda}) &= \frac{\exp(\lambda_k)}{\sum_{i=1}^n \exp(\lambda_i)} \sum_{j=1}^m \frac{\left(\varphi^{-1}\left(\frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_i)} \right) - y^{(j)} \right)}{\sqrt{\left(\varphi^{-1}\left(\frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_i)} \right) - y^{(j)} \right)^2 + \varepsilon^2}} \\
&\cdot (\varphi^{-1})' \left(\frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_i)} \right) \\
&\cdot \left(\varphi\left(x_k^{(j)}\right) - \frac{\sum_{i=1}^n \exp(\lambda_i)\varphi\left(x_i^{(j)}\right)}{\sum_{i=1}^n \exp(\lambda_i)} \right).
\end{aligned}$$

Assuming that $\mathbf{Z} = \mathbf{w}^T \varphi(\mathbf{X})$ and $\mathbf{w} = \exp(\boldsymbol{\lambda})/(\mathbf{1}^T \exp(\boldsymbol{\lambda}))$, we have:

$$\nabla E(\boldsymbol{\lambda}) = \mathbf{w} \cdot \left(\frac{\left(\phi^{-1}(\mathbf{Z}) - Y\right) \cdot (\varphi^{-1})'(\mathbf{Z})}{\sqrt{(\varphi^{-1}(\mathbf{Z}) - \mathbf{Y}) \cdot (\varphi^{-1}(\mathbf{Z}) - \mathbf{Y}) + \varepsilon^2}} \times \left(\varphi(\mathbf{X})^T - \mathbf{Z}\right) \right).$$

Remark 1.169. In Figure A.8 we provide an implementation of the aforementioned weight fitting procedure. It is based on the BFGS algorithm available via the `optim()` function in R. For testing purposes, we set up convergence criteria to be `reltol` $= 1e-16$, `maxiter` $= 10000$.

It is well-known that LAD optimization using nonlinear solvers does not guarantee that the output result is the global optimum: the BFGS algorithm may sometimes get stuck in a suboptimal solution or fail to converge in a predefined number of iterations.

For instance, suppose that $\varphi(x) = x^2$, $n = 5$, $m = 25$, $\varepsilon = 10^{-12}$ and that $\mathbf{X}$ and $\mathbf{Y}$ are generated randomly like in Example 1.167. The presented procedure gives median relative L_1 error (as compared to the optimal solution determined by the routine in Figure A.5) of $1\times \simeq 10^{-12}$ ($M = 10000$ MC iterations). On the other hand, the BFGS algorithm applied directly on an absolute value-based error function gives median relative error of $\simeq 6 \times 10^{-4}$. The 99%-quantiles are, respectively, around 3×10^{-7} and 0.5. Thus, the suggested approximation works far better than the direct approach.

Sometimes it may be advisable to run the optimization routine a few times, starting each time from a different initial point and then choose the best (in terms of L_1 error) solution. For instance, in the current experiment setting, using 10 trials reduces the median error of the "exact method" to $\simeq 6 \times 10^{-5}$, i.e., by a factor of 10. However, in the case of the approximate method, we did not get any significant improvement in terms of the median error, which already is close to the accuracy limits of computers' floating point arithmetic. Yet, the 99% quantile is now 10 times lower and we detected only 1 outlier case (instead of 11 – out of 10000) in which the relative error is greater than 1%. The 10-fold procedure failed to converge 76 (instead of 825) times within the presumed

`reltol` and `maxit` settings – in such circumstances one may try to rerun the BFGS algorithm from different random initial points until convergence criteria are satisfied.

1.6.5 Fitting weighted power means

Up to now we studied a case of weighted power means where φ was fixed (e.g., to an identity function which lead to the weighted arithmetic means). Let us now assume that we have a suspicion that φ might be an instance of some parametrized class of functions, e.g., $\varphi(x) = x^p, p > 1$. In other words, we are interested in fitting weighted power means to data.

In the case of the squared error, our task now becomes a bi-level optimization problem:

$$\text{minimize } E(p) \quad \text{w.r.t. } p$$

subject to $p \in [p_{\min}, p_{\max}]$ where $E(p)$ is a solution to:

$$\text{minimize } \sum_{j=1}^{m} \left(\sqrt[p]{\sum_{i=1}^{n} w_i \left(x_i^{(j)}\right)^p} - y^{(j)} \right)^2 \quad \text{w.r.t. } \mathbf{w}$$

subject to $\mathbf{w} \geq \mathbf{0}$, $\mathbf{1}^T\mathbf{w} = 1$. On a side note, Beliakov in [34] and Torra [444] consider a similar problem, however using the linearization technique. The L_1 error may be incorporated accordingly. Note that most often we observe that E is a quite well-behaving, unimodal function, therefore one-dimensional nonlinear solvers (like the Brent method [80]) may be utilized.

Example 1.170. Let us consider the data set generated as follows:

```
set.seed(132)
n <- 2
m <- 9
X <- t(matrix(runif(n*m, 0, 1), nrow=m))
p <- 2
realw <- runif(n)
realw <- realw/sum(realw)
Y <- matrix(as.numeric((t(X^p) %*% realw)^(1/p))+
     rt(m, 5)*0.05, ncol=m)
```

For least squares fitting we use the function in the mentioned Figure A.9. which relies on the already discussed solver for an optimization task given by Equation (1.53). The obtained L_1 and L_2 errors as a function of p are depicted in Figure 1.5. Here, the minimum was obtained for $p^* = 1.928388$, giving the total L_2 error of 0.1041785.

1.6.6 Determining generator functions of quasi-arithmetic means

What happens, however, if we would like to fit a weighted quasi-arithmetic mean to empirical data but we have no knowledge on how a φ generating function

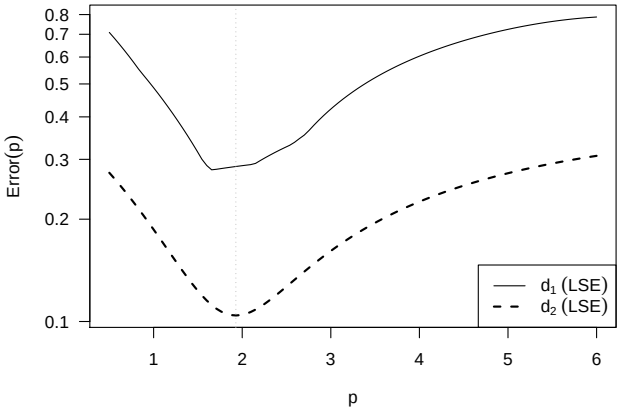


Figure 1.5. Approximation error (L_1 and L_2) as a function of p, see Example 1.170.

might be defined? In such a case, Beliakov et al. suggest to rely on the notion of spline functions, see, e.g., [32, 34, 49, 50]. Namely, we are now interested in a method that uses B-splines to construct the φ generator functions that are the basis of weighted quasi-arithmetic means.

Suppose that $p \geq 1$, $\mathbb{I} = [a, b]$ and let $\mathbf{t} = (t_1, \ldots, t_k)$ be an increasingly ordered *knot vector* of length k for some $k \geq 0$ such that $a < t_i < t_{i+1} < b$ for all $i \in [k]$. For brevity of notation we assume that $t_i = a$ for $i < 1$ and $t_i = b$ whenever $i > k$. Let us define *B-spline basis functions* for $j = 0, \ldots, p$ and $\theta \in [a, b]$ recursively as:

$$\begin{aligned} N^{\mathbf{t}}_{i,j}(\theta) &= \begin{cases} 1 & \text{if } \theta \in [t_{i-1}, t_i[, \\ 0 & \text{otherwise,} \end{cases} \qquad (j = 0) \\ N^{\mathbf{t}}_{i,j}(\theta) &= \frac{\theta - t_{i-1}}{t_{i+j-1} - t_{i-1}} N^{\mathbf{t}}_{i,j-1}(\theta) + \frac{t_{i+j} - \theta}{t_{i+j} - t_i} N^{\mathbf{t}}_{i+1,j-1}(\theta), \quad (j > 0) \end{aligned}$$

with convention $\cdot/0 = 0$.

Example 1.171. Figure 1.6 depicts B-spline basis functions $N^{\mathbf{t}}_{i-p,p}$ for $i = 1, \ldots, p+k+1$ in the case of $k = 2$ equidistant internal knots and $p = 1$ as well as $p = 3$ with $\mathbb{I} = [0, 1]$. Note that for all $\theta \in \mathbb{I}$ it holds $\sum_{i=1}^{p+k+1} N^{\mathbf{t}}_{i-p,p}(\theta) = 1$.

Let $\mathbf{v} \in \mathbb{I}^\eta$ be a vector of *control points*, where $\eta = p+k+1$. Then $B^{\mathbf{t}}_{\mathbf{v}} : \mathbb{I} \to \mathbb{I}$ given by:

$$B^{\mathbf{t}}_{\mathbf{v}}(\theta) = \sum_{i=1}^{\eta} v_i N^{\mathbf{t}}_{i-p,p}(\theta) \tag{1.56}$$

is a *nonperiodic B-spline of degree* p based on a knot vector $\mathbf{t}$, see, e.g., [420]. In particular, for $p = 1$ we get a piecewise linear function interpolating $(a, v_1), (t_1, v_2), \ldots, (t_k, v_{\eta-1}), (b, v_\eta)$. On the other hand, for $p = 3$ we get a cubic B-spline.

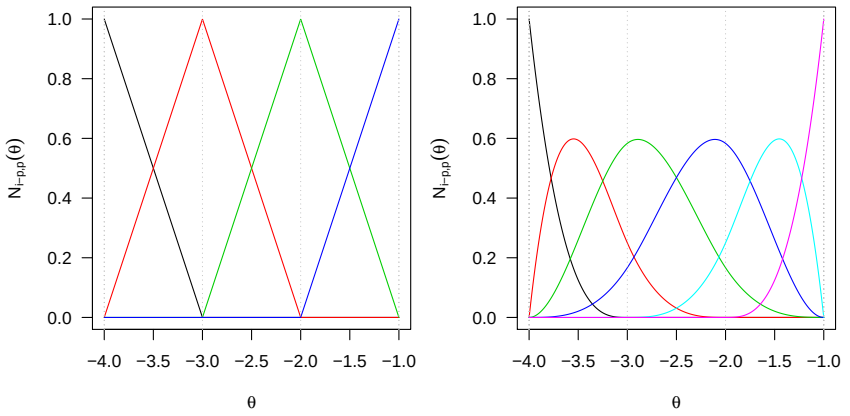


Figure 1.6. B-spline basis functions in the case of $k = 2$ equidistant internal knots and $p = 1$ (left) as well as $p = 3$ (right).

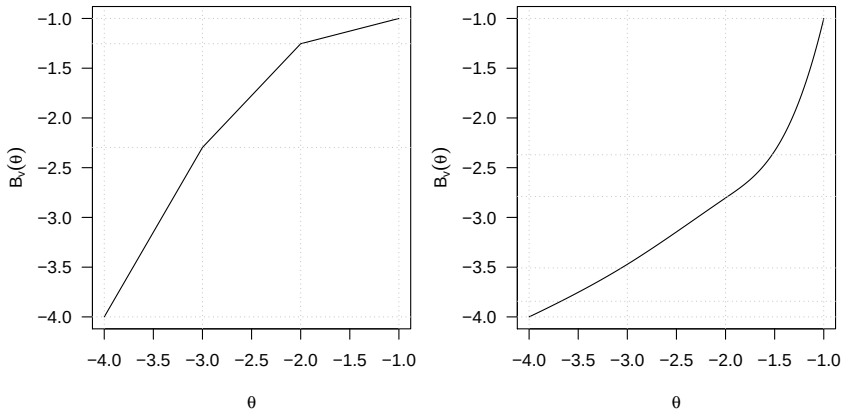


Figure 1.7. B-splines in the case of $k = 2$ equidistant internal knots and $p = 1$ (left, $\mathbf{v} = (0, 0.25, 0.8, 1)$) as well as $p = 3$ (right, $\mathbf{v} = (0, 0.1, 0.15, 0.2, 0.95, 1)$).

Remark 1.172. De Boor's algorithm (see, e.g., [309, 420]) may be used to compute B-splines. In R, this may be done using, for instance, the `splineDesign()` function from the splines package.

Example 1.173. Figure 1.7 depicts two exemplary B-splines: a piecewise linear one and a cubic one; we assume $\mathbb{I} = [0, 1]$.

Remark 1.174. The derivative of a B-spline of degree p is itself a B-spline of degree $p - 1$. It might be easily shown that if $\mathbf{v}$ is ordered increasingly, then its corresponding B-spline is strictly increasing. It is worth noting that if $v_1 = a$

and $v_\eta = b$, then $B^{\mathbf{t}}_{\mathbf{v}}$ is a function onto $\mathbb{I}$. Of course, if $p = 1$, then the inverse of an increasing B-spline is a B-spline of degree 1 (piecewise linear spline) as well. However, to the best of our knowledge, for $p > 1$ there are no analytic methods to determine $(B^{\mathbf{t}}_{\mathbf{v}})^{-1}$. Yet, the inverse may easily be computed numerically using, e.g., a root finding algorithm. Also, it may be approximated with other B-splines.

Assume that $\mathbf{t}$ is fixed (see, e.g., [243] and references therein for a discussion on knot selection) and that $\varphi(x) = B^{\mathbf{t}}_{\mathbf{v}}(x) = \sum_{i=1}^{\eta} v_i N^{\mathbf{t}}_{i-p,p}(x)$ for some increasing $\mathbf{v} \in \mathbb{I}^\eta$ such that $v_1 = a$ and $v_\eta = b$. If $\mathbf{w}$ is given a priori and we rely on the linearization technique (see page 95), our WQAMean fitting procedure may be expressed as:

$$\text{minimize} \sum_{j=1}^{m} \left(\sum_{k=1}^{\eta} v_k u_{j,k} \right)^2 \quad \text{w.r.t. } (v_2, \ldots, v_{\eta-1})$$

in the case of the squared error, or:

$$\text{minimize} \sum_{j=1}^{m} \left| \sum_{k=1}^{\eta} v_k u_{j,k} \right| \quad \text{w.r.t. } (v_2, \ldots, v_{\eta-1})$$

in the case of the absolute error, subject to:

$$\begin{aligned} v_2 &> a \\ v_i - v_{i-1} &> 0 \text{ for } i = 3, \ldots, \eta - 1 \\ v_{\eta-1} &< b, \end{aligned}$$

where:

$$u_{j,k} = \sum_{i=1}^{n} w_i N^{\mathbf{t}}_{k-p,p}(x_i^{(j)}) - N^{\mathbf{t}}_{k-p,p}(y^{(j)}),$$

see, e.g., [31, 37, 49]. If $\mathbf{w}$ is also unknown, then a two-stage optimization procedure may be used, see, e.g., [34, 37]. Alternatively, one may rely on a "global" optimization routine like CMA-ES [240]. Note that assuring that $v_1 < v_2 < \cdots < v_\eta$ may be done via reparametrization: one may use variables like v'_i with boundary constraints on $v'_i > 0$ for $i = 2, \ldots, \eta$, where $v_i = \sum_{j=1}^{n} v'_i$. In is also worth noting that Beliakov and James in [45] also considered B-splines fitting in the case of a LAD task and Bonferroni means.

1.6.7 A note on hierarchies of quasi-arithmetic means

Recall that in Example 1.86 we considered the case of feedforward neural networks, which were isomorphic to a hierarchy of quasi-arithmetic means.

It is well known that a neural network serves as a universal approximator: for instance, many successful applications of such machine learning algorithms were

reported in classification problems. To *train* a neural network, the Widrow-Hoff "backpropagation" (backward error propagation) algorithm, see, e.g., [467] may be used (among others) – it is based on stochastic gradient descent techniques; the updating algorithm is applied until weights no longer change significantly under the mean square error minimization criterion.

1.7 Aggregation on bounded posets

It turns out that in some intelligent systems and other applications, elements we aggregate are non-numeric or although they are represented as numbers, albeit cannot be treated as being defined on the so-called *interval scale*. In such a context operations like $+, -, \cdot, /$, as well as $\sqrt{\cdot}$, $\exp\cdot$, $\sin\cdot$ may not be meaningful at all.

In this section, we relax our (strong up to now) assumptions on the input domain and suppose that the aggregated elements may only be somehow ordered. This is the case of, for example, linguistic information: values of some attributes may be represented as labels like "low", "medium", "high" or "bad", "good", "excellent", etc., compare also the Zadeh *computing with words* methodology [488]. It is clear that here statements like "3·bad" or "warm+10" make no sense. This implies that most of the previously defined data fusion techniques, e.g., OWA and weighted averaging, must be replaced with some more elaborated solutions.

1.7.1 Basic order theory concepts

Assume that elements we aggregate come from a set P (possibly uncountable) and a preordering relation has been established. Recall that a *preorder* over P is a binary relation $\sqsubseteq \subseteq P \times P$ which is:

(a) *reflexive*, i.e., $(\forall p \in P)$ it holds $p \sqsubseteq p$,

(b) *transitive*, i.e., $(\forall p, q, r \in P)$ $p \sqsubseteq q$ and $q \sqsubseteq r \implies p \sqsubseteq r$.

A set P equipped with a preorder $\sqsubseteq$, i.e., $(P, \sqsubseteq)$, is called a *preordered set*.

Moreover, any *antisymmetric* preorder $\sqsubseteq$, that is, a binary relation such that $(\forall p, q \in P)$ if $p \sqsubseteq q$ and $q \sqsubseteq p$, then $p = q$, is called a *partial order* and then $(P, \sqsubseteq)$ is called a *poset* (partially ordered set). In such a case, we sometimes write $p \sqsubset q$ to indicate the fact that $p \sqsubseteq q$ and $p \neq q$.

Example 1.175. Let $P = \{\text{beautiful}, \text{rich}, \text{famous}, \text{wise}\}$. A decision maker introduces the following partial order $\sqsubseteq$ over P, expressing his/her "life desires":

$$\sqsubseteq = \left\{ \begin{array}{l} (\text{beautiful}, \text{beautiful}), (\text{rich}, \text{rich}), (\text{famous}, \text{famous}), (\text{wise}, \text{wise}), \\ (\text{beautiful}, \text{rich}), (\text{beautiful}, \text{famous}), (\text{rich}, \text{wise}), (\text{famous}, \text{wise}), \\ (\text{beautiful}, \text{wise}). \end{array} \right\},$$

Note that, actually, the pairs in the second row above are the most "informative". The elements in the first row are implied by reflexivity and in the third row – by transitivity. Also please notice that rich and famous are not comparable with $\sqsubseteq$.

Remark 1.176. If P is finite, then from the formal (syntactic) perspective each preordered set may be represented as a directed graph (there is a one-to-one correspondence between directed graphs and binary relations). Thus, instead of writing $p \sqsubseteq q$ we may presume that there is an edge from p to q, where $p, q \in P$. A simplified version of the poset (directed graph) from Example 1.175 may be depicted as in Figure 1.8.

What we see there is a *Hasse diagram*. An arrow (edge) from p to q, $p, q \in P$, means that $p \sqsubseteq q$. Loops, i.e., edges from each p to p itself, are not included in the diagram for readability. Moreover, please notice that edges implied by transitivity are also hidden. In other words, an ordering relation may be obtained from a Hasse diagram by calculating its reflexive and transitive closure. Also please observe that, e.g., the Warshall algorithm [463] may be used to find a transitive closure of a graph represented as a 0-1 matrix in $O(|P|^3)$ time. The opposite operation, transitive reduction, may be obtained by a method by Aho, Garey, and Ullman [6], who additionally showed that this problem is of the same computational complexity as that of finding the corresponding closure. It might be shown that both tasks may be efficiently solved via binary matrix multiplication, i.e., in at most $O(|P|^{2.3728639})$-time [308].

Additionally, a *total* partial order $\sqsubseteq$, i.e., such that $(\forall p, q \in P)$ it holds $p \sqsubseteq q$ or $q \sqsubseteq p$, is called a *linear order*.

Example 1.177. Let $P = \{\text{tiny}, \text{small}, \text{normal}, \text{large}, \text{huge}\}$ denote the set of TEX font sizes. We may establish a linear order $\sqsubseteq$ over P with the Hasse diagram below:

$$\text{tiny} \longrightarrow \text{small} \longrightarrow \text{normal} \longrightarrow \text{large} \longrightarrow \text{huge}.$$

By transitivity, we of course have $\text{small} \sqsubset \text{large}$, etc.

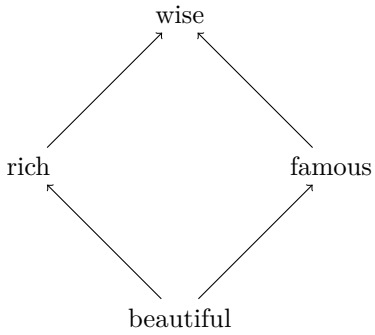


Figure 1.8. An illustration for Example 1.176.

Remark 1.178. If $(P, \sqsubseteq)$ is a finite chain, then it may be represented as a real interval I (with standard ordering of reals) by means of an order-preserving *utility function* $f : P \to I$, which is defined up to a strictly increasing bijection $\varphi : I \to I$, see, e.g., [337, 343] for discussion. For instance, in Example 1.177 f may be such that $f(\text{tiny}) = 1$, $f(\text{small}) = 2$, $f(\text{normal}) = 3$, $f(\text{large}) = 4$, $f(\text{huge}) = 5$.

Given a poset $(P, \sqsubseteq)$, if there exists $\underline{0} \in P$ for which $(\forall p \in P)$ it holds $\underline{0} \sqsubseteq p$, then we call such $\underline{0}$ *the least element* of P. Similarly, *the greatest element* of P is defined as $\overline{1} \in P$ such that $(\forall p \in P)$ we have $p \sqsubseteq \overline{1}$ (if it exists). $(P, \sqsubseteq, \underline{0}, \overline{1})$ is called a *bounded poset*, if the poset $(P, \sqsubseteq)$ has the least element $\underline{0}$ and the greatest element $\overline{1}$.

Example 1.179. In Example 1.175 we presented a bounded poset with $\underline{0} =$ beautiful and $\overline{1} =$ wise.

A *lattice* $(P, \sqsubseteq, \sqcap, \sqcup)$ is a poset in which every pair of elements has a unique infimum (meet, $\sqcap$, the greatest element of common lower bounds) and supremum (join, $\sqcup$, the smallest element of common upper bounds). If $\sqsubseteq$ is a linear order, then a lattice is called a *chain*.

A lattice $(P, \sqsubseteq, \sqcap, \sqcup)$ is called *distributive* whenever for all $p, q, r \in P$

$$p \sqcup (q \sqcap r) = (p \sqcup q) \sqcap (p \sqcup r) \tag{1.57}$$

or, equivalently,

$$p \sqcap (q \sqcup r) = (p \sqcap q) \sqcup (p \sqcap r), \tag{1.58}$$

which is exactly the same as requiring:

$$p \sqcap r = q \sqcap r \text{ and } p \sqcup r = q \sqcup r \implies p = q. \tag{1.59}$$

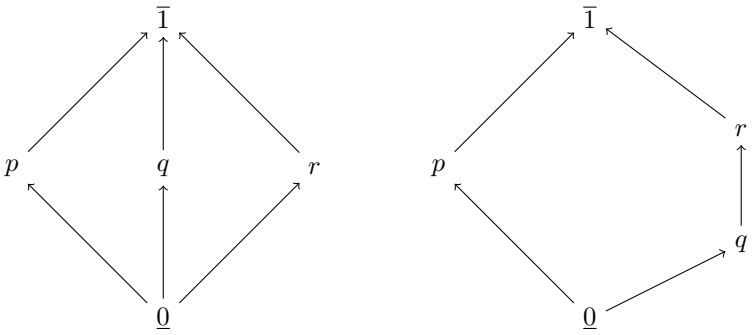
Additionally, it may be shown that a lattice is distributive if and only if none of its sublattices is isomorphic to any of the two simplest non-distributive lattices depicted in Figure 1.9.

Moreover, we call a lattice $(P, \sqsubseteq, \sqcap, \sqcup)$ *complete*, whenever every subset $P' \subseteq P$ has a unique supremum (denoted with $\bigsqcup P' = \bigsqcup_{p' \in P'} p'$) and infimum ($\bigsqcap P'$). Clearly, every complete lattice is bounded.

1.7.2 Aggregation functions on bounded posets

We have established the three most common scenarios, from the most to the least general:

1. bounded posets,
2. bounded lattices,
3. chains.



(a) diamond lattice M_3 (b) pentagon lattice N_5

Figure 1.9. The two simplest non-distributive lattices.

By the term *fusion function* we now mean any mapping $\mathsf{F}^{(n)} : P^n \to P$. In order to reintroduce the concept of an aggregation function, this time in a bounded poset setting, we follow the definition given, e.g., in [145].

Definition 1.180. Let $(P, \sqsubseteq, \underline{0}, \overline{1})$ be a bounded poset. A mapping $\mathsf{F}^{(n)} : P^n \to P$, is called an *aggregation function* if:

(a) $(\forall \mathbf{x}, \mathbf{y} \in P^n)$ if $(\forall i \in [n])\ x_i \sqsubseteq y_i$, then $\mathsf{F}^{(n)}(\mathbf{x}) \sqsubseteq \mathsf{F}^{(n)}(\mathbf{y})$,

(b) $\mathsf{F}^{(n)}(n * \underline{0}) = \underline{0}$, (lower boundary condition)

(c) $\mathsf{F}^{(n)}(n * \overline{1}) = \overline{1}$. (upper boundary condition)

Remark 1.181. Using the introduced notion, we may define extended aggregation functions like $\mathsf{F}^* : P^* \to P$ by assuming that for all n the restriction $\mathsf{F}^*|_{P^n}$ is an aggregation function.

Remark 1.182. Let $\mathbb{I} = [a, b]$ and $\leq$ denote the standard ordering of reals. In the case of the bounded chain $(\mathbb{I}, \leq, a, b, \wedge, \vee)$, the above definition coincides with the classical one as given in [230]. An example of such an aggregation function is the sample minimum. On the other hand, the arithmetic mean cannot be given as an instance of this class, as in its definition some "illegal" arithmetic operations occur.

Komorníková and Mesiar note in [293] that some properties of "ordinary" aggregation functions may be straightforwardly transformed to the case of fusion functions on bounded posets. This happens, e.g., in the case of:

— symmetry (see Definition 1.35),

— idempotency (Definition 1.24),

— associativity (Definition 1.114),

— decomposability (Definition 1.123),

— bisymmetry (Definition 1.126),

— annihilator (Definition 1.140) and neutral (Definition 1.141) element.

Internality is sometimes defined as (see [293, 382, 383]):

$$\mathsf{F}^{(n)}(x_1, \ldots, x_n) \in \{x_1, \ldots, x_n\} \tag{1.60}$$

or, if we act on a complete lattice, alternatively as (see [382]):

$$\bigsqcap_{i=1}^{n} x_i \sqsubseteq \mathsf{F}^{(n)}(x_1, \ldots, x_n) \sqsubseteq \bigsqcup_{i=1}^{n} x_i. \tag{1.61}$$

In this regard, the former one naturally arises when we require that a fusion function is comparison meaningful (preserves relative output order under any inputs' order automorphism) and the latter one stands for a basis of *means in the Ovchinnikov sense*.

Both cases may lead to undesired consequences if the aggregated elements are incomparable. Thus, in the following section we review an appealing proposal on how to solve this issue. As a side effect, we also present a categorization of aggregation functions on bounded posets.

1.7.3 Classes of fusion functions

Recall that we distinguished four main classes of nondecreasing fusion functions (see [169]):

— internal (averaging),

— conjunctive (`AND`-like, e.g., t-norms),

— disjunctive (`OR`-like, e.g., t-conorms),

— mixed.

On the interval scale, the distinction was based on their relationship to Min and Max. If we act on chains, we may replace Min and Max with inf and sup, respectively. Yet, on more general (bounded) posets the situation is somehow more complicated.

At this point, let us follow the classification proposed by Komorníková and Mesiar in [293], which was inspired by the notion of k-intolerance introduced in [339]. Given an aggregation function $\mathsf{F}^{(n)} : P^n \to P$, let:

$$\gamma(\mathsf{F}^{(n)}) = \inf\left\{\left|i : \mathsf{F}^{(n)}(\mathbf{x}) \sqsubseteq x_i\right| : \mathbf{x} \in P^n\right\}, \tag{1.62}$$

$$\sigma(\mathsf{F}^{(n)}) = \inf\left\{\left|i : x_i \sqsubseteq \mathsf{F}^{(n)}(\mathbf{x})\right| : \mathbf{x} \in P^n\right\}. \tag{1.63}$$

Definition 1.183 ([293]). We call $\mathsf{F}^{(n)} : P^n \to P$ *strongly conjunctive*, whenever $\mathsf{F}^{(n)} \in \gamma^{-1}(n)$. What is more, $\mathsf{F}^{(n)} : P^n \to P$ is *strongly disjunctive*, if $\mathsf{F}^{(n)} \in \sigma^{-1}(n)$.

Remark 1.184. In other words, $\mathsf{F}^{(n)}$ is strongly conjunctive, if for all $\mathbf{x} \in P^n$ and all $i \in [n]$ it holds $\mathsf{F}^{(n)}(\mathbf{x}) \sqsubseteq x_i$. Recall that on an interval scale we called $\mathsf{F}^{(n)}$ conjunctive, whenever for all $\mathbf{x}$ it held $\mathsf{F}^{(n)}(\mathbf{x}) \leq \mathsf{Min}(\mathbf{x})$, or equivalently $\mathsf{F}^{(n)}(\mathbf{x}) \leq x_i$ for all $i \in [n]$.

Definition 1.185 ([293]). We call $\mathsf{F}^{(n)} : P^n \to P$ *weakly conjunctive*, whenever $\mathsf{F}^{(n)} \in \bigcup_{i=1}^{n-1} \gamma^{-1}(i)$. Moreover, we say that $\mathsf{F}^{(n)} : P^n \to P$ is *weakly disjunctive*, if $\mathsf{F}^{(n)} \in \bigcup_{i=1}^{n-1} \sigma^{-1}(i)$.

Based on the above notion, we may introduce the concept of an averaging aggregation function.

Definition 1.186 ([293]). We call $\mathsf{F}^{(n)} : P^n \to P$ *weakly averaging*, whenever it is weakly conjunctive or weakly disjunctive. Moreover, it is *strongly averaging* if it is both weakly conjunctive and weakly disjunctive.

Intuitively, a weakly averaging function outputs values that are greater than or less than some elements we aggregate, but definitely not greater than or less than all such elements in every possible case. Each weakly/strongly averaging function is idempotent. Moreover, if P is a bounded lattice, the only strongly conjunctive (disjunctive) and idempotent aggregation function is the Min (respectively, Max).

Having said that, aggregation functions on bounded posets may be classified as:

(a) weakly averaging $(\bigcup_{i=1}^{n-1} \gamma^{-1}(i) \cup \bigcup_{i=1}^{n-1} \sigma^{-1}(i))$,

(b) strongly conjunctive $(\gamma^{-1}(n))$,

(c) strongly disjunctive $(\sigma^{-1}(n))$,

(d) mixed $(\gamma^{-1}(0) \cap \sigma^{-1}(0))$.

This is what was called in [293] a weak classification. Its strong version assumes that the class of weakly averaging functions may additionally be considered as consisting of aggregation functions that are either:

(a′) strongly averaging $(\bigcup_{i=1}^{n-1} \gamma^{-1}(i) \cap \bigcup_{i=1}^{n-1} \sigma^{-1}(i))$,

(a″) weakly conjunctive but not weakly disjunctive $(\bigcup_{i=1}^{n-1} \gamma^{-1}(i) \setminus \bigcup_{i=1}^{n-1} \sigma^{-1}(i))$,

(a‴) weakly disjunctive but not weakly conjunctive
$(\bigcup_{i=1}^{n-1} \sigma^{-1}(i) \setminus \bigcup_{i=1}^{n-1} \gamma^{-1}(i))$.

Both classification schemes are *complete* in the sense that any function falls exactly into one category.

Remark 1.187. If we act on a bounded chain, then the weak and strong classification in the Komorníková-Mesiar [293] sense and the "classical" Dubois-Prade classification [169] are equivalent.

1.7.4 Idempotent fusion functions

Please note that triangular norms and conorms on a bounded poset $(P, \sqsubseteq, \underline{0}, \overline{1})$ may be defined via a straightforward generalization of the case presented in Definitions 1.143 and 1.146, see [138, 139]. This is because none of their sine qua non properties are specific to $[0, 1]$ and a natural linear order $\leq$. Moreover, e.g., Karaçal and Mesiar [267] studied uninorms on bounded lattices.

Nevertheless, our main focus in this book is on the study of fusion functions that are at least idempotent. Various authors also translate some well-known classical averaging aggregation functions to the framework of aggregation on posets. Here are a few examples.

Example 1.188. Let $P = \{p_1, \ldots, p_k\}$ be a finite set equipped with a total ordering relation $\sqsubseteq$ and assume that $p_1 \sqsubseteq \cdots \sqsubseteq p_k$. Moreover, let $\mathbf{w}$ be a weighting vector of length n and suppose that w_i is a weight corresponding to p_i. Then the Yager [480] *weighted median* is defined as p_l with the smallest possible l such that $\sum_{i=1}^{l} w_i \geq 0.5$. Input data of this kind naturally occur when elements come from a multiset over a totally ordered set (see Example 1.81). It is easily seen that the input median is idempotent, averaging, symmetric, and monotone. Noteworthily, an iterative algorithm for weights fitting was also provided in this case, see [480, Section 4].

Example 1.189. Let $P = \{p_0, p_1, \ldots, p_k\}$ be a finite set equipped with a total ordering relation $\sqsubseteq$ and suppose $p_0 \sqsubseteq \cdots \sqsubseteq p_n$. The *linguistic OWA operator* introduced by Herrera, Herrera-Viedma, and Verdegay in [245] is generated by a weighting vector $\mathbf{w}$ and is defined for a given $\mathbf{x} \in P^n$, $n \geq 2$ as follows. Assume that $\sigma \in \mathfrak{S}_{[n]}$ is such that $x_{\sigma(1)} \sqsubseteq \cdots \sqsubseteq x_{\sigma(n)}$. Then:

$$\mathsf{LOWA}_{\mathbf{w}}(\mathbf{x}) = C^n(\mathbf{w}, \mathbf{x}),$$

where the "convex combination" of elements in P operator [143] C^n is defined for $n > 2$ recursively as:

$$C^n(\mathbf{w}, \mathbf{x}) = C^2\Big((1 - w_n, w_n),$$

$$\left(C^{n-1}\left(\left(\frac{w_1}{1-w_n},\dots,\frac{w_{n-1}}{1-w_n}\right),\left(x_{\sigma(1)},\dots,x_{\sigma(n-1)}\right)\right),x_{\sigma(n)}\right),$$

and for $n=2$ – under assumption $k\geq j\geq i\geq 0$ and $w+w'=1$ – as:

$$C^2((w,w'),(p_j,p_i))=p_{k\wedge(i+\mathrm{round}(w(j-i)))}.$$

We see that in fact we map elements in P to the set of nonnegative integers. It can be shown that the linguistic OWA operator is monotonic, averaging, idempotent, and symmetric. The above idea was enhanced by Godo and Torra [225] who introduced the so-called qualitative OWA operators. Such operators utilize the notion of the t-norm instead of the C function. Moreover, Kolesárová, Mayor, and Mesiar in [290] study a different approach for constructing weighted ordinal means based on divisible discrete t-norms.

Example 1.190. Lizasoain and Moreno [326] note that the original OWA operator for $\mathbf{x}\in[0,1]^n$:

$$\mathsf{OWA}_{\mathbf{w}}(\mathbf{x})=\sum_{i=1}^{n}w_ix_{(i)}$$

generated by a weighting vector $\mathbf{w}$ may be rewritten as:

$$\mathsf{OWA}_{\mathbf{w}}(\mathbf{x})=\mathsf{S}_{\text{Ł}}\left(\mathsf{T}_{\mathrm{P}}(w_1,x_{(1)}),\dots,\mathsf{T}_{\mathrm{P}}(w_n,x_{(n)})\right)$$

with assumption $\mathsf{S}_{\text{Ł}}(w_1,\dots,w_n)=1$, where T_{P} and $\mathsf{S}_{\text{Ł}}$ denote the product t-norm and Łukasiewicz t-conorm, respectively. Assuming that we act on a complete lattice and substituting arbitrary t-norms and t-conorms valid there for T_{P} and $\mathsf{S}_{\text{Ł}}$, we may introduce OWA-like lattice operators as long as we are able to order the input observations. Of course, if we are on a chain, this task is trivial. In other cases, the authors propose to follow the approach of, e.g., Ovchinnikov [382], and compute an OWA operator on inputs like y_i (instead of $x_{(i)}$), where:

$$\begin{aligned}
y_1 &= x_1\sqcap\cdots\sqcap x_n,\\
&\vdots\\
y_i &= \bigsqcup_{\{j_1,\dots,j_{n-i+1}\}\subseteq[n]} x_{j_1}\sqcap\cdots\sqcap x_{j_{n-i+1}},\\
&\vdots\\
y_{n-1} &= \bigsqcup_{\{j_1,j_2\}\subseteq[n]} x_{j_1}\sqcap x_{j_2},\\
y_n &= x_1\sqcup\cdots\sqcup x_n,
\end{aligned}$$

which fulfill:

$$y_1\sqsubseteq y_2\sqsubseteq\cdots\sqsubseteq y_n.$$

It can be noted that if we are on a chain, then $y_i=x_{(i)}$. All OWA-like lattice operators are idempotent.

1.7.5 Lattice polynomial functions

Let us generalize the notion of a (weighted) lattice polynomial function, see Equation (1.32), to the case of a complete distributive lattice $(P, \sqsubseteq, \sqcap, \sqcup, \underline{0}, \overline{1})$. Assume that $\bigsqcup_{x \in \emptyset} x = \underline{0}$ and $\bigsqcap_{x \in \emptyset} x = \overline{1}$. Lattice polynomial functions are formed as expressions that consist of variables in P which are linked by the $\sqcap$, $\sqcup$ lattice operations applied in any order, see [62].

Example 1.191. Here is an exemplary lattice polynomial function of four variables: $\mathsf{F}^{(n)}(x_1, x_2, x_3, x_4) = (x_1 \sqcap x_2) \sqcup (x_3 \sqcap x_4)$.

Definition 1.192. The class of n-argument *lattice polynomial functions* (n-LPF) from P^n to P is defined by applying the following rules finitely many times:

(a) $\mathsf{F}^{(n)}(x_1, \ldots, x_n) = x_i$ is an n-LPF for any $i \in [n]$,

(b) If $\mathsf{F}^{(n)}$ and $\mathsf{G}^{(n)}$ are n-LPFs, then $\mathsf{F}^{(n)} \sqcap \mathsf{G}^{(n)}$ and $\mathsf{F}^{(n)} \sqcup \mathsf{G}^{(n)}$ are n-LPFs.

Remark 1.193. Each lattice polynomial function is nondecreasing with respect to $\sqsubseteq$.

Example 1.194. A ternary median on a bounded distributive lattice is given by:

$$\begin{aligned} \mathsf{Median}^{(3)}(x_1, x_2, x_3) &= (x_1 \sqcap x_2) \sqcup (x_2 \sqcap x_3) \sqcup (x_3 \sqcap x_1) \\ &= (x_1 \sqcup x_2) \sqcap (x_2 \sqcup x_3) \sqcap (x_3 \sqcup x_1). \end{aligned}$$

It turns out that each n-LPF may be written in a simpler form. We have what follows, see [62].

Proposition 1.195. *Let $\mathsf{F}^{(n)}$ be an n-LPF. Then there exist $k, l \geq 1$ and families $A_1, \ldots, A_k, B_1, \ldots, B_l$ of nonempty subsets of $[n]$ such that:*

$$\mathsf{F}^{(n)}(x_1, \ldots, x_n) = \bigsqcup_{j=1}^{k} \bigsqcap_{i \in A_j} x_i = \bigsqcap_{j=1}^{l} \bigsqcup_{i \in B_j} x_i. \tag{1.64}$$

Example 1.196. Fix $t \in [n]$. Let $k = \binom{n}{t}$ and $\mathcal{A} = \{A_1, \ldots, A_k\} = \{\{i_1, \ldots, i_t\} \subseteq [n]\}$. If we are on a chain, then for any $\mathbf{x} \in P^n$ it holds that $x_{(n-t+1)} = \bigsqcup_{j=1}^{k} \bigsqcap_{i \in A_j} x_i$, i.e., the tth order statistic, see [382, 383]. In fact, see [337], any symmetric n-LPF on a chain is an order statistic.

The class of weighted lattice polynomial functions has been generalized by Marichal in [341].

Definition 1.197. The class of n-argument *weighted lattice polynomial functions* (n-WLPF) from P^n to P is defined by applying the following rules finitely many times:

(a) $\mathsf{F}^{(n)}(x_1, \ldots, x_n) = x_i$ is an n-WLPF for any $i \in [n]$,

(b) $\mathsf{F}^{(n)}(x_1, \ldots, x_n) = p$ is an n-WLPF for any $p \in P$,

(c) If $\mathsf{F}^{(n)}$ and $\mathsf{G}^{(n)}$ are n-WLPFs, then $\mathsf{F}^{(n)} \sqcap \mathsf{G}^{(n)}$ and $\mathsf{F}^{(n)} \sqcup \mathsf{G}^{(n)}$ are n-WLPFs.

As an analogue of Proposition 1.195, we have the following result.

Proposition 1.198. *Let $\mathsf{F}^{(n)}$ be an n-WLPF. Then there exist $k, l \geq 1$, constants $a_1, \ldots, a_k, b_1, \ldots, b_l \in P$, and families $A_1, \ldots, A_k, B_1, \ldots, B_l$ of nonempty subsets of $[n]$ such that:*

$$\mathsf{F}^{(n)}(x_1, \ldots, x_n) = \bigsqcup_{j=1}^{k} \left(a_j \sqcap \bigsqcap_{i \in A_j} x_i \right) = \bigsqcap_{j=1}^{l} \left(b_j \sqcup \bigsqcup_{i \in B_j} x_i \right). \tag{1.65}$$

Interestingly, it turns out that n-ary weighted lattice polynomial functions may also be represented as below.

Proposition 1.199. *Let $\mathsf{F}^{(n)}$ be an n-WLPF. Then there exist set functions $\alpha, \beta : 2^{[n]} \to P$ such that:*

$$\mathsf{F}^{(n)}(x_1, \ldots, x_n) = \bigsqcup_{S \subseteq [n]} \left(\alpha(S) \sqcap \bigsqcap_{i \in S} x_i \right) = \bigsqcap_{S \subseteq [n]} \left(\beta(S) \sqcup \bigsqcup_{i \in S} x_i \right). \tag{1.66}$$

It can be shown that, e.g., $\alpha(S) = \beta([n] \setminus S)$ in the above equation. An n-WLPF formulated as above is said to be either in disjunctive (left) or conjunctive (right) normal form.

Example 1.200. By [341, Corollary 13], see also [335], $\mathsf{F}^{(n)}$ is a Sugeno integral if and only if $\mathsf{F}^{(n)}$ is an idempotent n-WLPF. And this happens if and only if $\mathsf{F}^{(n)}(n * \underline{0}) = \underline{0}$ and $\mathsf{F}^{(n)}(n * \overline{1}) = \overline{1}$, i.e., it is endpoint preserving.

Proposition 1.201 ([131]). *$\mathsf{F}^{(n)}$ is a symmetric n-WLPF if and only if it can be represented in a disjunctive or conjunctive normal form (see Proposition 1.199) with $\alpha(S)$ and $\beta(S)$ being cardinality-based, i.e., solely functions of $|S|$.*

1.8 Aggregation on a nominal scale

Having been given a space of objects on which an ordering relation is defined is a quite comfortable situation. It is even more pleasant, if we can rely on this assumption in such a way that we may require that a fusion function must preserve such an order. Unfortunately, in some practical applications we do not have as much as that.

Let us assume that the elements to be aggregated are defined on a nominal scale. That is, there is a finite set, $\Sigma = \{\mathtt{a}_1, \ldots, \mathtt{a}_k\}$, called an *alphabet*, on which only an equivalence relation, $=$, is defined. Each element of Σ is called a *character*.

Example 1.202. In molecular biology and bioinformatics (among others), we may assume $\Sigma = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$, i.e., a set consisting of the primary nucleobases: adenine, cytosine, guanine, and thymine, respectively. Here, we may also be interested in the protein alphabet, which is of cardinality 20.

Example 1.203. Σ may also be the set of code points covered by the Unicode standard. The Universal Coded Character Set defines more than 110,000 characters (letters, numbers, symbols, etc.) from most languages, scripts, and locales. Alternatively, it may be the set of characters covered by the ASCII (see Table 2.4) or ISO-8859-1 standard. Note that even though encoding standards define mappings between sets of characters and integers (on which a natural linear order exists), it does not mean that we obtain anything more than just a nominal scale here.

Remark 1.204. In the R programming language, there is a special data type to store information on a nominal scale called `factor`. Such objects are represented as integer vectors with a special attribute, *levels*, which is used to decode the numeric indices into string labels.

```
x <- factor(c("a", "g", "c", "a", "t", "g"))
print(x)
## [1] a g c a t g
## Levels: a c g t
table(x)
## a c g t
## 2 1 2 1
unclass(x)        # internal representation
## [1] 1 3 2 1 4 3       # integer indices
## attr(,"levels")
## [1] "a" "c" "g" "t"   # decoding scheme
```

Example 1.205. We may also assume that $\Sigma = \{\mathtt{0}, \mathtt{1}\}$ is a set of bits, i.e., *b*inary dig*it*s.

It turns out that fusion functions defined on objects on a nominal scale, although useful in practical applications, are not too "mathematically interesting". Perhaps the only sensible family of metrics we may define in the current setting is given by:

$$\mathfrak{d}_c(\mathtt{a},\mathtt{b}) = c\mathbf{1}(\mathtt{a} \neq \mathtt{b}), \tag{1.67}$$

where $\mathtt{a},\mathtt{b} \in \Sigma$, which for $c = 1$ is in fact the Hamming distance on Σ^1, see also Section 2.7.

Given $\mathbf{x} \in \Sigma^n$, $\mathtt{a} \in \Sigma$ such that:

$$a = \arg\min_{\mathtt{a}\in\Sigma} \sum_{i\in[n]} \mathfrak{d}_c(x_i, \mathtt{a}).$$

is equivalent to the *mode* of $\mathbf{x}$, i.e., the most frequently occurring observation in $\mathbf{x}$, see also Remark 1.30. Note that the solution to the above equation may be non-unique. Nevertheless, assuming that $\Sigma = \{\mathtt{a}_1, \ldots, \mathtt{a}_k\}$, we may introduce a fusion function, e.g., like:

$$\mathsf{Median}^{(n)}_{\mathfrak{d}_c}(x_1, \ldots, x_n) = \mathtt{a}_j,$$

where $j = \min\{j : \mathtt{a}_j = \arg\min_{\mathtt{a}\in\Sigma} \sum_{i\in[n]} \mathfrak{d}_c(x_i, \mathtt{a})\}$, which now is well-defined.

Remark 1.206. Assuming that $\Sigma = \{1, 2, \ldots, k\}$, there are a few possible approaches to determine a mode:

— a bucket-sort like algorithm requires $O(k + n)$ time,

— the elements may be sorted with the radix sort algorithm, which requires $O(n \log k)$ time,

— a hash-table-based procedure requires amortized $O(n)$ time,

and so on.

The introduced fusion function is:

— symmetric,

— idempotent,

— such that $\mathsf{Median}^{(n)}_{\mathfrak{d}_c}(\mathbf{x}) = \mathsf{Median}^{(n)}_{\mathfrak{d}_c}(\mathbf{y})$ where $y_i \in \{x_i, \mathsf{Median}^{(n)}_{\mathfrak{d}_c}(\mathbf{x})\}$,

— stable, see [212, 405], i.e.,

$$\mathsf{Median}^{(n+1)}_{\mathfrak{d}_c}(x_1, \ldots, x_n, \mathsf{Median}^{(n)}_{\mathfrak{d}_c}(x_1, \ldots, x_n)) = \mathsf{Median}^{(n)}_{\mathfrak{d}_c}(x_1, \ldots, x_n).$$

Example 1.207. A *weighted mode* is a fusion function which minimizes:

$$\mathtt{a} = \arg\min_{\mathtt{a}\in\Sigma} \sum_{i\in[n]} w_i \mathfrak{d}_c(x_i, \mathtt{a}).$$

for some weighting vector $\mathbf{w}$. This tool is used in a class of machine learning algorithms for classification called *ensemble methods*. For instance, in the so-called *bagging* (bootstrap averaging), see, e.g., [78], w_i is defined as $\alpha_i / \sum_j \alpha_j$, where α_i is the ith classifier's accuracy. The famous *random forest* algorithm is based on the very same idea, compare [78].

Please note that the case of aggregating observations on a nominal scale becomes much more challenging when we shall consider d- or arbitrary-dimensional data.

Chapter 2

Aggregation of multivariate data

Let us focus on the task dealing with aggregation of n objects in a d-dimensional space X^d, where this time $d > 1$. This is a case of, e.g., real vectors in $\mathbb{R}^d$, Cartesian products of d identical bounded posets, as well as d-digits binary or nucleobase sequences.

For fixed d, consider a fusion function $\mathsf{F} : (X^d)^n \to X^d$ that aims to aggregate a set of n objects $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in X^d$. By using this mapping we obtain a single object from the set X^d. In other words, F is such that:

$$\mathsf{F}\left(\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_d^{(1)} \end{bmatrix}, \ldots, \begin{bmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_d^{(n)} \end{bmatrix}\right) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_d \end{bmatrix}. \tag{2.1}$$

Equivalently, we may conceive F as a function acting on a $d \times n$ matrix:

$$\mathbf{X} = [\mathbf{x}^{(1)} \ \mathbf{x}^{(2)} \ \cdots \ \mathbf{x}^{(n)}].$$

From now on we assume that all vectors are column vectors. Note that in data analysis, $\mathbf{x}^{(i)}$ is often called an *observation* – it designates an object or experimental unit. On the other hand, $x_j^{(i)}$ denotes the result of measuring the jth variable or feature (such as temperature, weight, velocity, etc.) of the ith observation (e.g., a person, autonomous vehicle, spatial location).

First we shall review the task of real vectors' fusion from the perspective of aggregation theory. In the consecutive sections, we significantly extend the results presented in [208].

2.1 Aggregation of real vectors

Most of the aggregation methods reviewed in this section come from areas like computational statistics and geometry. Therefore, here we shall assume that $X = \mathbb{R}$.

Example 2.1. Let us take any three non-colinear points in $\mathbb{R}^2$. Even in such a simple case there are many useful ways to aggregate a triad, see the triangle center problem [263, 272, 273]. Most often this issue is conceptualized by using the so-called *triangle center function*, see [72], which is a homogeneous real-valued function of a triangle's side lengths. Thus, when rewritten in terms of vertex coordinates, this leads us to a fusion function which is – among others – rotation and scale equivariant (see below). Among the most well-known triangle centers we find the centroid, in-, circum-, and ortho-center. What is interesting, C. Kimberling's *Encyclopedia of Triangle Centers* (available online at `http://faculty.evansville.edu/ck6/encyclopedia/ETC.html`) as of December 10, 2015 lists, names, and characterizes over 8781 such aggregation methods.

As we know from Chapter 1, in classical aggregation theory, we mostly focus on the $d = 1$ case. Recall that the notion of a mean (internal aggregation function) $\mathsf{F} : \mathbb{R}^n \to \mathbb{R}$, may be used to determine the "most typical observation" among a given set of values. We know that identifying the sine qua non conditions that F should fulfill in order to be useful in particular applications is very important, as the class of all fusion functions is of course too broad. Following the axiomatic framework by Kolmogorov and Nagumo, see, e.g., [87, 292, 370] and Remark 1.44, we could require the fulfillment of at least the three following properties:

— symmetry,

— nondecreasingness, and

— internality.

Let us extend them in such a way that they are valid for any d.

Symmetry. The first property is the least problematic one. We may simply assume that for any $\sigma \in \mathfrak{S}_{[n]}$ it holds:

$$\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathsf{F}(\mathbf{x}^{(\sigma(1))}, \ldots, \mathbf{x}^{(\sigma(n))}) \tag{2.2}$$

It turns out that the easiest and perhaps the most natural approach to extend the other two properties is to apply them in a *componentwise manner*.

Nondecreasingness. First of all, note that the ordering structure on $\mathbb{R}$ may easily be extended to $\mathbb{R}^d$ by determining the so-called *product order*. The partial order $\leq_d$ is defined in such a way that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ we have:

$$\mathbf{x} \leq_d \mathbf{y} \text{ if and only if } (\forall i \in [d])\ x_i \leq y_i. \tag{2.3}$$

This leads us to the concept of $\leq_d$ (componentwise)-nondecreasingness. Such an approach is often used when the topic of aggregation on posets is explored, see, e.g., [91, 138, 293], and also Section 2.6.

Definition 2.2. A fusion function $\mathsf{F} : (\mathbb{R}^d)^n \to \mathbb{R}^d$ is $\leq_d$-*nondecreasing* whenever for all $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)} \in \mathbb{R}^d$ such that $\mathbf{x}^{(i)} \leq_d \mathbf{y}^{(i)}$ for all $i = 1, \ldots, n$ it holds $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) \leq_d \mathsf{F}(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)})$.

Internality. On the other hand, componentwise internality may be defined as follows.

Definition 2.3. A fusion function $\mathsf{F} : (\mathbb{R}^d)^n \to \mathbb{R}^d$ is *componentwise internal* if for all $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in \mathbb{R}^d$ it holds:

$$\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) \in \left[\bigwedge_{i=1}^{n} x_1^{(i)}, \bigvee_{i=1}^{n} x_1^{(i)} \right] \times \cdots \times \left[\bigwedge_{i=1}^{n} x_d^{(i)}, \bigvee_{i=1}^{n} x_d^{(i)} \right]. \tag{2.4}$$

Basically, above we deal with the bounding (hyper)rectangle of a given set of input points.

Here are two exemplary fusion functions that fulfill symmetry as well as componentwise monotonicity and internality.

Definition 2.4. The componentwise extension of the arithmetic mean is given by:

$$\mathsf{CwAMean}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \begin{bmatrix} \frac{1}{n} \sum_{i=1}^{n} x_1^{(i)} \\ \vdots \\ \frac{1}{n} \sum_{i=1}^{n} x_d^{(i)} \end{bmatrix}.$$

This fusion function is also called the *centroid* (barycenter, geometric center) of a set of points. This notion is crucial, e.g., in the definition of the k-means [331] clustering algorithm.

On the other hand, the following mapping is sometimes used, see [432], as a robust estimate of a multidimensional probability distribution's median.

Definition 2.5. The componentwise extension of the sample median is defined as:

$$\mathsf{CwMedian}(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)}) = \begin{bmatrix} \mathsf{Median}\left(x_1^{(1)},\ldots,x_1^{(n)}\right) \\ \vdots \\ \mathsf{Median}\left(x_d^{(1)},\ldots,x_d^{(n)}\right) \end{bmatrix}.$$

Both functions are examples of componentwise extensions of an internal aggregation function $\mathsf{G}:\mathbb{R}^n\to\mathbb{R}$. The induced fusion function $\mathsf{CwG}:(\mathbb{R}^d)^n\to\mathbb{R}^d$ combines each data dimension independently. Thus, we have:

$$\mathsf{CwG}\left(\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_d^{(1)} \end{bmatrix},\ldots,\begin{bmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_d^{(n)} \end{bmatrix}\right) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_d \end{bmatrix} = \begin{bmatrix} \mathsf{G}(x_1^{(1)},\ldots,x_1^{(n)}) \\ \mathsf{G}(x_2^{(1)},\ldots,x_2^{(n)}) \\ \vdots \\ \mathsf{G}(x_d^{(1)},\ldots,x_d^{(n)}), \end{bmatrix}. \tag{2.5}$$

It is easily seen that if G is nondecreasing in each variable, then for $\mathbf{x}^{(1)}\leq_d \mathbf{y}^{(1)},\ldots,\mathbf{x}^{(n)}\leq_d\mathbf{y}^{(n)}$, we get $\mathsf{CwG}(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)})\leq_d\mathsf{CwG}(\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(n)})$. Thus, CwG is $\leq_d$-nondecreasing.

Even more generally, we may of course consider the class of *decomposable* (as named, e.g., in [293]) fusion functions:

$$\mathsf{F}_{\mathsf{G}_1,\ldots,\mathsf{G}_d}\left(\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_d^{(1)} \end{bmatrix},\ldots,\begin{bmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_d^{(n)} \end{bmatrix}\right) = \begin{bmatrix} \mathsf{G}_1(x_1^{(1)},\ldots,x_1^{(n)}) \\ \mathsf{G}_2(x_2^{(1)},\ldots,x_2^{(n)}) \\ \vdots \\ \mathsf{G}_d(x_d^{(1)},\ldots,x_d^{(n)}), \end{bmatrix}, \tag{2.6}$$

where $\mathsf{G}_i:\mathbb{R}^n\to\mathbb{R}$, $i=1,\ldots,d$. However, we shall note that in the case of such a class of fusion functions, no interactions between different dimensions are taken into account explicitly.

Thus, in practice more intricate fusion functions are used. Let us note that, see [208], the following data aggregation tools – well known in data analysis – do not fulfill the componentwise monotonicity. We will inspect them in much greater detail later on, so now let us only provide their basic definitions.

Definition 2.6. The (Euclidean) *1-median* is a point $\mathbf{y}$ such that:

$$\mathsf{1median}_{\mathfrak{d}_2}(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)}) = \operatorname*{arg\,min}_{\mathbf{y}\in\mathbb{R}^d}\frac{1}{n}\sum_{i=1}^{n}\mathfrak{d}_2(\mathbf{x}^{(i)},\mathbf{y}), \tag{2.7}$$

where $\mathfrak{d}_2$ is again the Euclidean distance.

Example 2.7. Also 1-median is not componentwise monotone. Take $d = 2$, $n = 3$, and $\mathbf{x}^{(1)} = [0, 0]^T$, $\mathbf{x}^{(2)} = [1, -5]^T$, $\mathbf{x}^{(3)} = [20, 1]^T$. We have $\mathsf{1median}_{\mathfrak{d}_2}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}) \simeq [1.961, -2.305]^T$. However, when we take $\mathbf{x}'^{(3)} = \mathbf{x}^{(3)} + [1980, 1]^T$, then we get $\mathsf{1median}_{\mathfrak{d}_2}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}'^{(3)}) \simeq [1.946, -3.351]^T <_2 [1.961, -2.305]^T$.

Definition 2.8. The *Euclidean 1-center* (smallest enclosing ball radius) is given by:

$$\mathsf{1center}_{\mathfrak{d}_2}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \arg\min_{\mathbf{y} \in \mathbb{R}^d} \bigvee_{i=1}^{n} \mathfrak{d}_2(\mathbf{x}^{(i)}, \mathbf{y}), \tag{2.8}$$

where $\mathfrak{d}_2$ is the Euclidean metric.

Example 2.9. Euclidean 1-center is not componentwise monotone. Consider $n = 3$ and $d = 2$ with $\mathbf{x}^{(1)} = [1, -1]^T$, $\mathbf{x}^{(2)} = [-1, 1]^T$, $\mathbf{x}^{(3)} = [-\sqrt{2}, 0]^T$. We have $\mathsf{1center}_{\mathfrak{d}_2}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}) = [0, 0]^T$. Letting $\mathbf{x}'^{(1)} = \mathbf{x}^{(1)} + [3, 0]^T$ we get $\mathsf{1center}_{\mathfrak{d}_2}(\mathbf{x}'^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}) \approx [1.3, -0.5]^T \not\geq_2 [0, 0]^T$.

Moreover, Tukey [451] introduced the concept of *the halfplane location depth* of $\mathbf{y}$ relative to a given set of points in $\mathbb{R}^d$. It is the smallest number of points contained in any closed halfhyperplane with boundary line through $\mathbf{y}$. In other words:

$$\mathsf{tdepth}_d(\mathbf{y}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \min_{\mathbf{u} \in \mathbb{R}^d, |\mathbf{u}|=1} |\{i : \mathbf{u}^T \mathbf{x}^{(i)} \geq \mathbf{u}^T \mathbf{y}\}|. \tag{2.9}$$

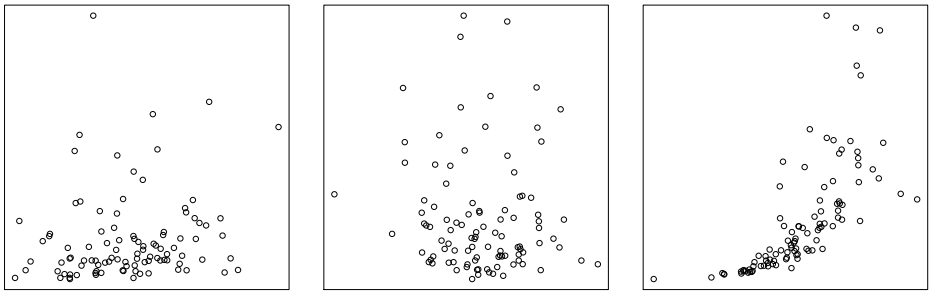
Observe that the deepest point in $d = 1$ generalizes the concept of the median, at least for odd n. Therefore, a deepest value in higher dimensions can be thought of as a multidimensional median.

Definition 2.10. The center of gravity of the deepest halfplane location depth region is called the *Tukey median*, $\mathsf{TkMedian}$.

Example 2.11. Tukey median is not componentwise monotone. Consider $n = 4$ and $d = 2$ with $\mathbf{x}^{(1)} = [0, 0]^T$, $\mathbf{x}^{(2)} = [1, 0]^T$, $\mathbf{x}^{(3)} = [1, 1]^T$, and $\mathbf{x}^{(4)} = [0, 1]^T$. We have $\mathsf{TkMedian}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}) = [0.5, 0.5]^T$. Letting $\mathbf{x}'^{(4)} = \mathbf{x}^{(4)} + [1, 0]^T$ we get $\mathsf{TkMedian}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}'^{(4)}) = [2/3, 1/3]^T \not\geq_2 [0.5, 0.5]^T$.

Remark 2.12. To model a d-dimensional data set $\mathbf{X}$ we may make use of marginal cumulative distribution functions $F_1, \ldots, F_d$ and a d-dimensional copula C (see Definition 1.147), which describes the interdependence between individual data dimensions. This is because, according to the famous Sklar theorem [430], see also [373], whatever the joint cumulative distribution function H of $(X_1, \ldots, X_d)$ is, i.e.:

$$H(x_1, \ldots, x_d) = \mathrm{P}(X_1 \leq x_1, \ldots, X_d \leq x_d),$$



(a) Gaussian ($\varrho = 0.4$) (b) Product (c) Clayton ($\theta = 4$)

Figure 2.1. Effects of choosing different copulas if marginal cumulative distribution functions are $F_1 = \mathrm{N}(0, 1), F_2 = \mathrm{Exp}(0.1)$.

there always exists (unique if H is continuous) $\mathsf{C}, F_1, \ldots, F_d$ such that:

$$H(x_1, \ldots, x_d) = \mathsf{C}(F_1(x_1), \ldots, F_d(x_d)).$$

Such a description is also useful if random variates generation is needed. A procedure for obtaining a single random vector in $\mathbb{R}^d$ may thus look as follows:

1. Generate $(Y_1, \ldots, Y_d) \sim \mathsf{C}$ (note that copula C is in fact a cumulative distribution function on the unit hypercube);
2. Return $(F_1^{-1}(Y_1), \ldots, F_d^{-1}(Y_d)) \sim H$ as result.

Here is an exemplary R code that uses the copula [485] package to generate a sample of $n = 100$ ($d = 2$)-dimensional points using the Clayton copula with parameter $\theta = 4$, $\mathsf{C}(u, v) = \left(0 \vee u^{-\theta} + v^{-\theta} - 1\right)^{-1/\theta}$, and $F_1 = \mathrm{N}(0, 1)$ (standard normal), $F_2 = \mathrm{Exp}(0.1)$ (an exponential distribution).

```
n <- 100
d <- 2
C <- copula::claytonCopula(dim=d, param=4)
Finv <- list( # marginal c.d.f.s (inverses)
   function(y) qnorm(y, 0, 1), function(y) qexp(y, 0.1)
)

X <- t(copula::rCopula(n, C))
for (i in 1:d) X[i,] <- Finv[[i]](X[i,])
```

Refer to Figure 2.1 for an illustration of effects of choosing different copulas.

2.2 Equivariance to geometric transforms

Instead of focusing on monotonicity and internality, researchers in such fields as computational statistics and geometry most often consider equivariances with

respect to specific classes of geometrical transformations. This is in line with the aforementioned fact that the necessity of the notion of monotonicity is being put into question in the classical framework too, see, e.g., [43, 93]. In the $d = 1$ case this property seems quite natural and moreover it simplifies the way the analytic results are derived. However, the situation is much different in higher dimensions.

Namely, one might be interested in finding a fusion function F which fulfills for all input vectors:

— *translation equivariance*: for all $\mathbf{t} \in \mathbb{R}^d$,

$$\mathsf{F}(\mathbf{x}^{(1)} + \mathbf{t}, \ldots, \mathbf{x}^{(n)} + \mathbf{t}) = \mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) + \mathbf{t},$$

— *uniform scale equivariance*: for all $s > 0$,

$$\mathsf{F}(s\mathbf{x}^{(1)}, \ldots, s\mathbf{x}^{(n)}) = s\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}),$$

— *d-scale equivariance*: for all $\mathbb{R}^d \ni \mathbf{s} > \mathbf{0}$,

$$\mathsf{F}(\mathbf{s}\mathbf{x}^{(1)}, \ldots, \mathbf{s}\mathbf{x}^{(n)}) = \mathbf{s}\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}).$$

— *orthogonal equivariance*: for all orthogonal matrices $\mathbf{A} \in \mathbb{R}^{d\times d}$,

$$\mathsf{F}(\mathbf{A}\mathbf{x}^{(1)}, \ldots, \mathbf{A}\mathbf{x}^{(n)}) = \mathbf{A}\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}),$$

and/or

— *affine equivariance*: for all matrices $\mathbf{A} \in \mathbb{R}^{d\times d}$ of full rank and all $\mathbf{t} \in \mathbb{R}^d$,

$$\mathsf{F}(\mathbf{A}\mathbf{x}^{(1)} + \mathbf{t}, \ldots, \mathbf{A}\mathbf{x}^{(n)} + \mathbf{t}) = \mathbf{A}\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) + \mathbf{t}.$$

Affine equivariance implies translation, uniform scale, d-scale, and orthogonal equivariance. Recall that with the notation convention used throughout this book, e.g., affine equivariance may be written as $\mathsf{F}(\mathbf{A}\mathbf{X}+\mathbf{t}) = \mathbf{A}\mathsf{F}(\mathbf{X})+\mathbf{t}$. Thus, an affine equivariant fusion function is independent of the chosen coordinate system. It is a very strong property, so let us start our discussion with simpler transformations. Also, we cover equivariance to similarity transforms, which includes the translation, uniform scale, and orthogonal equivariance.

We are interested in exploring basic facts about different types of equivariances, as well as different ways to modify a given mapping (especially one that is a componentwise extension of a classical aggregation function) so that it obeys the most important properties.

2.2.1 Translation and scale equivariance

It turns out that, given any fusion function F, it is quite easy to transform it in such a way that it becomes translation and uniform scale equivariant.

Proposition 2.13. *Let* $\mathsf{F}, \mathsf{G} : (\mathbb{R}^d)^n \to \mathbb{R}^d$ *be two fusion functions and assume that* G *is translation equivariant. Then* F' *given by:*

$$\mathsf{F}'(\mathbf{X}) = \mathsf{F}(\mathbf{X} - \mathsf{G}(\mathbf{X})) + \mathsf{G}(\mathbf{X})$$

is translation equivariant.

Note that G is often set to be the componentwise mean.

Proposition 2.14. *Let* $\mathsf{g} : (\mathbb{R}^d)^n \to \mathbb{R}$ *be a function such that* $\mathsf{g}(s\mathbf{X}) = s\mathsf{g}(\mathbf{X})$ *with* $\mathsf{g}(\mathbf{X}) \neq 0$ *for all nondegenerate* $\mathbf{X}$*. Assuming that* $\mathsf{F} : (\mathbb{R}^d)^n \to \mathbb{R}^d$ *is a fusion function, we have that* F' *given by:*

$$\mathsf{F}'(\mathbf{X}) = \mathsf{g}(\mathbf{X})\mathsf{F}\left(\frac{1}{\mathsf{g}(\mathbf{X})}\mathbf{X}\right)$$

is uniform scale equivariant for all nondegenerate $\mathbf{X}$*.*

In practice, we may set, e.g., $\mathbf{g}(\mathbf{X}) = \sqrt{\sum_{i=1}^n \sum_{j=1}^n \mathfrak{d}_2^2(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}$, which may be thought of as a multivariate extension of the sample standard deviation, see Section 5.2.

A quite similar result may be provided for the d-scale equivariance. Notably, each d-scale equivariant fusion function is also uniform scale equivariant.

Remark 2.15. Translation and d-scale equivariance is highly useful in the practice of data analysis, as one often standardizes the input variables:

$$x_j^{(i)} \mapsto \frac{x_j^{(i)} - \mathsf{AMean}(x_j^{(1)}, \ldots, x_j^{(n)})}{\mathsf{SD}(x_j^{(1)}, \ldots, x_j^{(n)})},$$

where AMean and SD stand for the arithmetic mean and standard deviation, respectively, which are applied on the jth coordinate, $j = 1, \ldots, d$.

Here is a result concerning componentwise extensions of interval scale equivariant univariate fusion functions, see Definition 1.54. On a side note, recall that we stated in Section 1.5 that the only quasi-arithmetic mean that is interval scale equivariant is the arithmetic mean.

Proposition 2.16. *If* $\mathsf{G} : \mathbb{R}^n \to \mathbb{R}$ *is such that* $\mathsf{G}(s\mathbf{x} + t) = s\mathsf{G}(\mathbf{x}) + t$ *for all* $\mathbf{x} \in \mathbb{R}^n, t \in \mathbb{R}, s > 0$*, then its componentwise extension* CwG *is translation and* d*-scale equivariant.*

2.2.2 Orthogonal equivariance

Some machine learning algorithms (such as principal component analysis, see Remark 2.20) assume that the data points may freely be rotated. Orthogonal

equivariance implies equivariance to all possible rotations of input points, reflections against the axes, and their combinations. The discussed equivariance type – especially together with translation equivariance – may be important if we do not wish to be dependent on the choice of a coordinate system.

Recall that $\mathbf{A}$ is an *orthogonal matrix* whenever it holds $\mathbf{A}\mathbf{A}^T = \mathbf{A}^T\mathbf{A} = \mathbf{I}$ or, equivalently, $\mathbf{A}^T = \mathbf{A}^{-1}$.

Remark 2.17. If $\mathbf{A}$ is orthogonal, then $|\det \mathbf{A}| = 1$ and columns of $\mathbf{A}$ are orthogonal unit vectors – they form an orthonormal basis of the Euclidean space $\mathbb{R}^d$. An $\mathbf{A}$-based transformation is *unitary*, i.e., it preserves the dot product of vectors. Thus, it preserves the Euclidean distance between two points (it is an isometry of the Euclidean space).

Generating random orthogonal matrices. Methods for random generation of orthogonal matrices may be used, e.g., for empirically testing whether a fusion function is orthogonal equivariant. Let $\mathcal{O}(d)$ denote the group of orthogonal $d \times d$ matrices.

Following [150], we may be interested in a uniform distribution on $\mathcal{O}(d)$ with respect to the Haar measure, see [238]. In other words, a random matrix $\mathbf{A}$ is uniformly distributed if $\mathrm{P}(\mathbf{A} \in \mathcal{A}) = \mathrm{P}(\mathbf{A} \in \mathbf{\Gamma}\mathcal{A})$, for any $\mathcal{A} \subset \mathcal{O}(d)$ and $\mathbf{\Gamma} \in \mathcal{O}(d)$.

For $d = 2$, a random matrix may be generated by considering $\vartheta \sim \mathrm{U}[0, 2\pi[$ and $b \sim \mathrm{U}\{-1, 1\}$ and then taking:

$$\mathbf{A} = \begin{bmatrix} \cos\vartheta & \sin\vartheta \\ -b\sin\vartheta & b\cos\vartheta \end{bmatrix}. \tag{2.10}$$

One way to generate a random orthogonal matrix for $d > 2$ is to produce a $d \times d$ matrix with i.i.d. elements following a standard normal distribution. Then, by applying the Gram-Schmidt orthogonalization algorithm on such a matrix, we get a desired object, see [176, page 234] for a proof. This gives an $O(d^3)$ algorithm, but in practice its implementation characterizes itself with slow performance. For this reason, we may rather want to use the following procedure.

Algorithm 2.18. *[150, Section 3] To generate a random orthogonal $d \times d$ matrix for given $d > 2$ proceed as follows:*

1. *Generate a random orthogonal 2×2 matrix $\mathbf{A}^{(2)}$, see Equation (2.10).*
2. *For $i = 3, 4, \ldots, d$ do:*
   - 2.1. *Let $\mathbf{v} \in \mathbb{R}^i$ be a randomly generated vector distributed uniformly on a unit i-sphere; for that we may generate $\mathbf{z} = (z_1, \ldots, z_i)$ i.i.d. $\mathrm{N}(0, 1)$ and set $\mathbf{v} := \mathbf{z}/\|\mathbf{z}\|_2$, see [345];*
   - 2.2. *Let $\mathbf{x} := (\mathbf{e}^{(i)} - \mathbf{v})/\|\mathbf{e}^{(i)} - \mathbf{v}\|_2$, where $\mathbf{e}^{(i)} = (1, 0, 0, \ldots, 0) \in \mathbb{R}^i$;*

2.3. Set:

$$\mathbf{A}^{(i)} := \left(\mathbf{I}^{(i)} - 2\mathbf{x}\mathbf{x}^T\right) \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & & & \\ \vdots & & \mathbf{A}^{(i-1)} & \\ 0 & & & \end{bmatrix},$$

where $\mathbf{I}^{(i)}$ *is the diagonal* $i \times i$ *matrix.*

3. Return $\mathbf{A}^{(d)}$ *as result.*

By carefully setting vector/matrix multiplication order in Step 2.3. we may get $O(d^3)$ time complexity, see Figure A.11 for an exemplary C++ implementation.

Orthogonal equivariant componentwise fusion functions. A special class of orthogonal projections consists of a kind of rotation combined with reflection. We take $\mathbf{A} = \mathbf{I}_\sigma$ for some $\sigma \in \mathfrak{S}_{[d]}$, i.e., an identity matrix with permuted rows. The equivariance with respect to such a transformation is the same as requiring that for each $\mathbf{X} \in (\mathbb{R}^d)^n$ it holds:

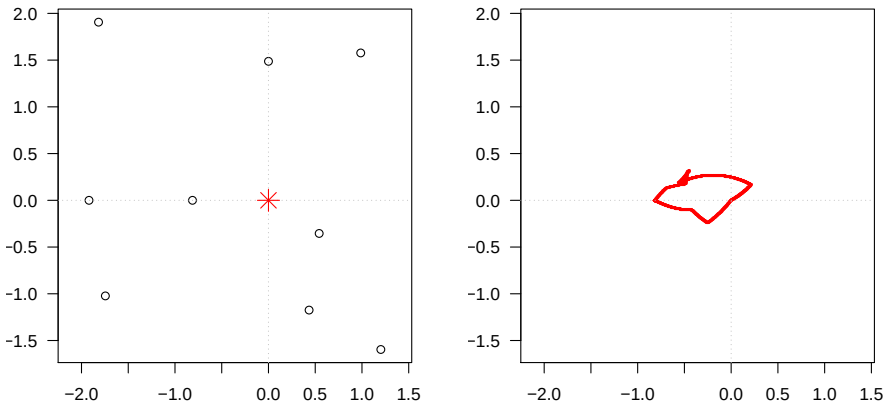
$$\mathsf{F}\left(\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_d^{(1)} \end{bmatrix}, \dots, \begin{bmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_d^{(n)} \end{bmatrix}\right) = \mathsf{F}\left(\begin{bmatrix} x_{\sigma(1)}^{(1)} \\ x_{\sigma(2)}^{(1)} \\ \vdots \\ x_{\sigma(d)}^{(1)} \end{bmatrix}, \dots, \begin{bmatrix} x_{\sigma(1)}^{(n)} \\ x_{\sigma(2)}^{(n)} \\ \vdots \\ x_{\sigma(d)}^{(n)} \end{bmatrix}\right)_{\sigma^{-1}}. \quad (2.11)$$

Thus, it is also a kind of symmetry (intuitively, a "vertical" one, as opposed to the componentwise symmetry discussed above). This easily leads us to the following result concerning componentwise extensions of unidimensional fusion functions.

Proposition 2.19. *If* $\mathsf{F}_{\mathsf{G}_1,\dots,\mathsf{G}_d}$ *is an orthogonal equivariant componentwise extension of* $\mathsf{G}_i : \mathbb{R}^n \to \mathbb{R}$, $i \in [d]$, *then necessarily it is a componentwise fusion function: there exists* G *such that* $\mathsf{CwG} = \mathsf{F}_{\mathsf{G}_1,\dots,\mathsf{G}_d}$. *Moreover, it necessarily holds that* $\mathsf{G}(x_1, \dots, x_n) = -\mathsf{G}(-x_1, \dots, -x_n)$.

We already noted that CwAMean is an orthogonal equivariant componentwise fusion function. However, the above necessary conditions are not sufficient: it turns out that the componentwise median, CwMedian, is not orthogonal equivariant.

Interestingly, even if we are given a non-orthogonal equivariant fusion function, we may *orthogonalize* it. Below we explain two particularly appealing orthogonalization methods, which may be used in the case of, e.g., the componentwise median, CwMedian, see Figure 2.2.



(a) A sample data set and its componentwise median $\mathsf{CwMedian}(\mathbf{X})$ at $(0, 0)$.

(b) Componentwise medians' trace over all rotations of the input data set, $\mathbf{A}^{-1}\mathsf{CwMedian}(\mathbf{AX})$.

Figure 2.2. Componentwise median and its dependence on the choice of a coordinate system.

Orthomedian. The orthomedian by Grubel [234] is an interesting instance of the concept of orthogonalization, originally applied on the componentwise median. Basically, it is the averaged median of all orthogonally transformed versions of the input data set. As the group $\mathcal{O}(d)$ of orthogonal $d \times d$ matrices is compact, we may introduce a fusion function:

$$\mathsf{OrMedian}(\mathbf{X}) = \int_{\mathcal{O}(d)} \mathbf{A}^{-1}\mathsf{CwMedian}(\mathbf{AX})\, d\mathbf{A}, \tag{2.12}$$

which is orthogonal equivariant (by construction) and additionally translation and uniform scale equivariant (but not d-scale equivariant). Interestingly, it is no longer $\leq_d$-nondecreasing, so this new property is introduced at some cost. The idea behind orthogonalization is quite general and may be applied in the case of other fusion functions as well.

One may (and should) ask how the orthogonal median may be computed. The above integral may of course be approximated via some Monte Carlo quadrature scheme. In such a case, random matrices sampled uniformly from $\mathcal{O}(d)$ can be generated (see Algorithm 2.18). However, this is computationally demanding and we observe a quite slow rate of convergence (e.g., for $d = 2$ we need at least 1000 MC iterations to get satisfiable results for a few dozen of points).

Another approach is to consider a set of N points on a unit d-hypersphere, $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(N)}$. Then the orthomedian may be approximated, see [234, Section 5], by:

$$\mathsf{OrMedian}(\mathbf{X}) \simeq d\mathsf{CwAMean}(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)}),$$

where:

$$\mathbf{y}^{(i)} = \mathsf{Median}\left(\mathbf{a}^{(i)T}\mathbf{x}^{(1)}, \ldots, \mathbf{a}^{(i)T}\mathbf{x}^{(n)}\right)\mathbf{a}^{(i)}.$$

The points on the hypersphere can be sampled randomly, but this process has an even slower convergence rate than the above-mentioned one. It is best to rely on a quasi-Monte Carlo approach and sample the points uniformly. This is easy for $d = 2$. In higher dimensions, however, the problem, at least for arbitrary N, becomes quite difficult. It is because we have to solve the (hyper)Sphere Packing (László Fejes Tóth's) problem, see [125], which concerns the task of placing N points on a d-dimensional hypersphere so as to maximize the minimal distance (or equivalently the minimal angle) between them. Such a task may be treated with a stochastic optimization routine (e.g., simulated annealing) or using an algorithm proposed in, e.g., [328]. Note that the probed points may be tabulated and stored for later use.

Before moving to the second orthogonalization method, let us briefly recall a statistical procedure called principal component analysis.

Remark 2.20. Principal component analysis (PCA) uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables, see [241, Section 3.4 and Section 14.5]. Let:

$$\mathbf{X}_c = \mathbf{X} - \mathsf{CwAMean}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$$

be a centered version of $\mathbf{X}$. Then the sample covariance matrix is given by $\mathbf{S} = \mathbf{X}_c\mathbf{X}_c^T/n \in \mathbb{R}^{d\times d}$. Let us take the eigendecomposition of:

$$\mathbf{X}_c\mathbf{X}_c^T = \mathbf{V}\mathbf{D}^2\mathbf{V}^T.$$

This may be obtained by taking the singular value decomposition (SVD):

$$\mathbf{X}_c^T = \mathbf{U}\mathbf{D}\mathbf{V}^T,$$

where $\mathbf{U}$ is an $n \times n$ orthogonal matrix, $\mathbf{D}$ is an $n \times d$ diagonal matrix with nonnegative elements, and $\mathbf{V}$ is a $d \times d$ orthogonal matrix, see the LAPACK [12] library routine `DGESDD`. The eigenvectors $\mathbf{v}^{(i)}$ are called *principal component directions* of $\mathbf{X}_c$. The first principal component direction $\mathbf{v}^{(1)}$ has the property that $\mathbf{z}^{(1)} = \mathbf{X}_c^T\mathbf{v}^{(1)}$ has the largest sample variance, d_1^2/n among all normalized linear combinations of $\mathbf{X}_c$'s rows. Subsequent principal components have maximum variance subject to being orthogonal to the earlier ones.

SVD-based orthogonalization. Given a unidimensional fusion function, $\mathsf{G} : \mathbb{R}^n \to \mathbb{R}$ such that $(\forall x_i)\ \mathsf{G}(x_1, \ldots, x_n) = -\mathsf{G}(-x_1, \ldots, -x_n)$, here is a simple way to orthogonalize its componentwise extension, CwG. Assuming that the SVD of $(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X}))^T = \mathbf{U}\mathbf{D}\mathbf{V}^T$ and knowing that $\mathbf{V} = \mathbf{V}^{T^{-1}}$, we may set

$$\mathsf{OrG2}(\mathbf{X}) = \mathbf{V}\mathsf{CwG}\left(\mathbf{V}^T(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X}))\right) + \mathsf{CwAMean}(\mathbf{X}). \tag{2.13}$$

It is easily seen that in such a way we obtain not only an orthogonal equivariant fusion function, but also one that is translation equivariant. Note that the

condition $\mathsf{G}(x_1, \ldots, x_n) = -\mathsf{G}(-x_1, \ldots, -x_n)$ is crucial, as the $\mathbf{U}, \mathbf{V}$ matrices might be ambiguous: the singular vectors are only defined up to sign; if we change the sign of a left singular vector, an equivalent SVD decomposition may be obtained by changing the sign of the corresponding right vector.

Here, if G is nondecreasing, then the resulting fusion function is nondecreasing with respect to the direction that has the maximal variance (and other directions that are orthogonal to it and also maximize the remaining variance).

Remark 2.21. Let $\mathsf{OrMedian2}$ be a SVD-orthogonalized version of the componentwise median for the case $d = 2$. Given $\mathbf{x}^{(1)} = (1, 1), \mathbf{x}^{(2)} = (1, -1), \mathbf{x}^{(3)} = (-1, -1), \mathbf{x}^{(4)} = (-1, 1)$, we have $\mathsf{OrMedian2}(\ldots) = (0, 0)$. Now letting $\mathbf{x}'^{(1)} = \mathbf{x}^{(1)} + (0, 2)$, we get $\mathsf{F}(\ldots) \approx (-0.25, 0.07)$. Thus, $\mathsf{OrMedian2}$ is not componentwise monotone.

2.2.3 Equivariance to similarity transforms

The class of similarity transforms includes translation, uniform scaling in each direction, rotation, and reflection. Equivariance to similarity transforms can be conceived as a "lightweight" version of the corresponding property with regard to affine transforms.

For any $\|\cdot\|$ matrix norm, if $\mathbf{A} \in \mathbb{R}^{d \times d}$ is a nondegenerate matrix, and $\mathbf{t} \in \mathbb{R}$, then $(\mathbf{A}, \mathbf{t})$ represents a similarity transform, whenever $\frac{1}{\|\mathbf{A}\|}\mathbf{A}$ is orthogonal.

Let us go back to the above-derived SVD-based componentwise fusion function orthogonalization scheme. For a given $\mathsf{G} : \mathbb{R}^n \to \mathbb{R}$, under the assumption that $\mathbf{UDV}^T$ is the SVD decomposition of $(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X}))^T$, and by noting that for any $s > 0$ we have:

$$s(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X})) = \mathbf{U}(s\mathbf{D})\mathbf{V}^T$$

we can define a similarity transform-equivariant fusion function as:

$$\mathsf{SimG}(\mathbf{X}) = \|\mathbf{D}\|{\mathbf{V}^{-1}}^T \mathsf{CwG}\left(\frac{1}{\|\mathbf{D}\|}\mathbf{V}^T(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X}))\right) + \mathsf{CwAMean}(\mathbf{X}),$$

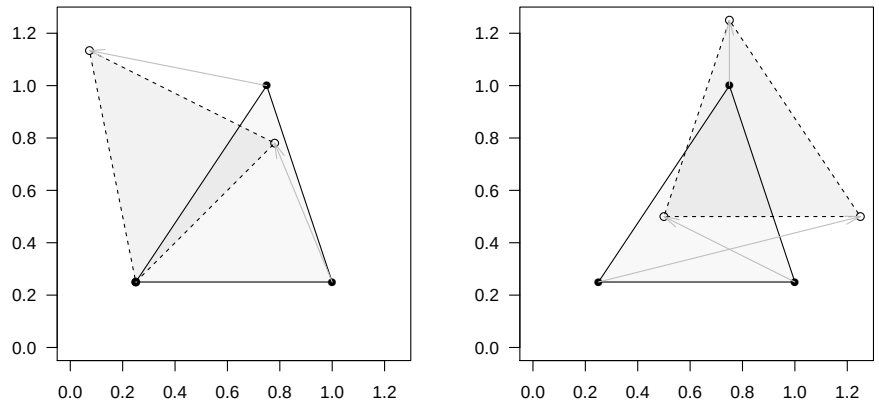
or, alternatively:

$$\mathsf{SimG}'(\mathbf{X}) = \mathbf{V}\mathbf{D}^T \mathsf{CwG}(\mathbf{U}^T) + \mathsf{CwAMean}(\mathbf{X}).$$

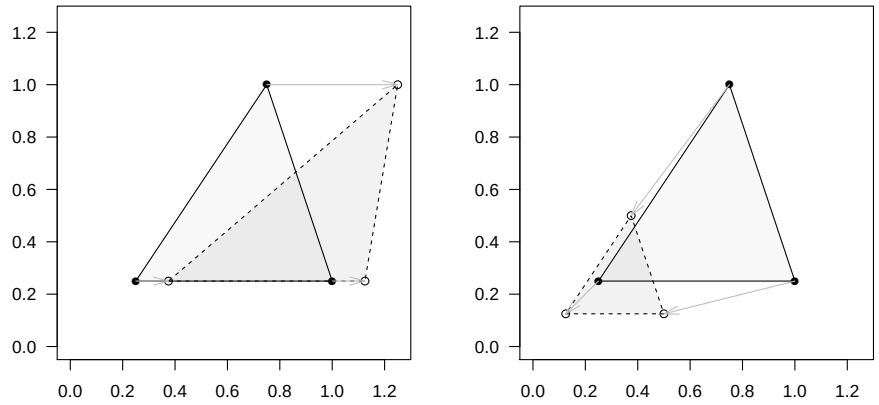
2.2.4 Affine equivariance

An affine transformation is a map that preserves hyperplanes: ratios of Euclidean distances of points lying on a straight line remain the same. Every linear transformation is affine, but not every affine transformation is linear.

Example 2.22. Table 2.1 lists exemplary affine transformations in $\mathbb{R}^2$. Among them we find, e.g., translation, scaling, rotation, shear mapping, reflection, and also any of their compositions. Figure 2.3 depicts some affine mappings.



(a) Translation, rotation, translation. (b) Reflection against OY, translation.



(c) Horizontal shear. (d) Scaling.

Figure 2.3. Exemplary affine transformations in $\mathbb{R}^2$.

Remark 2.23. Note that we require $\det \mathbf{A} \neq 0$. Otherwise, transformations such as projections onto OX and OY axes would also be included in our discussion. Yet, classically they are omitted as they lead to "too drastic" data loss.

It turns out that if affine equivariance is important to us, then we should be interested in rather "complex" fusion functions (see also Proposition 2.19). This is because of the following fact characterizing affine equivariant componentwise fusion functions.

Proposition 2.24. *A continuous and bounding-box internal fusion function* CwG*, being a componentwise extension of* $\mathsf{G} : \mathbb{R}^n \to \mathbb{R}$*, is affine equivariant if and only if* G *is a weighted arithmetic mean.*

The above result follows from the fact that G must necessarily be additive,

compare Theorem 1.133. Thus, the only componentwise symmetric, internal, continuous, and affine equivariant fusion function is formed by extending the arithmetic mean.

Among non-componentwise fusion functions that are affine equivariant we find, e.g., the Tukey median, TkMedian.

Remark 2.25. It turns out that an affine transformation $(\mathbf{A}, \mathbf{t})$ may be expressed using a single $(d+1)\times(d+1)$ square matrix. For that, the so-called *homogeneous coordinate system*, introduced by A.F. Möbius, is typically used. In order to do so, we first construct the *augmented matrix*:

$$\mathbf{B} = \left[\begin{array}{ccc|c} & \mathbf{A} & & \mathbf{t} \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Then, instead of operating on vectors like $\mathbf{x} \in \mathbb{R}^d$, we rather consider $[\mathbf{x}\ 1]^T \in \mathbb{R}^{d+1}$, i.e., a version of the original inputs with an additional coordinate equal to 1 added. In such a way:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{t}$$

may now be written as:

$$\left[\begin{array}{c} \mathbf{y} \\ \hline 1 \end{array}\right] = \left[\begin{array}{ccc|c} & \mathbf{A} & & \mathbf{t} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{c} \mathbf{x} \\ \hline 1 \end{array}\right].$$

Example 2.26. The homogeneous coordinate system is very common in 3D computer graphics, especially in games. This is the case of, e.g., First Person Perspective (FPP) shooters (like *Doom*, *Wolfenstein*, *Duke Nukem 3D*, *Quake*, or *Counter-Strike*) or flight simulators. Figure 2.4 gives a screenshot of an untextured terrain mesh in an exemplary 3D world simulation.

In such a setting, an agent is most often represented as a point $(0, 0, 0)$ and faces towards the $(1, 0, 0)$ vector. Here, the translation vector $\mathbf{t}$ may designate the current position of an agent. The affine $\mathbf{A}$ matrix provides the direction in which it looks. This is often provided by a composition of 3 rotation matrices given via the Euler angles – roll (OX rotation γ; unused in FPP shooters), pitch (OY rotation β, look up/down), and yaw (OZ rotation α, turn left/right):

$$\begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix}.$$

It is worth noting that modern graphics cards take advantage of homogeneous coordinates when a programmer implements vector and matrix algebra (e.g., via vertex shaders). OpenGL and Direct3D libraries allow for efficient data processing with 4-element (`float`) registers.

Table 2.1. Exemplary affine transformations in $\mathbb{R}^2$.

description	**transformation** $\mathbf{x} \mapsto \mathbf{A}\mathbf{x} + \mathbf{t}$	
Translation	$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$\mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$
Rotation by ϑ	$\mathbf{A} = \begin{bmatrix} \cos\vartheta & -\sin\vartheta \\ \sin\vartheta & \cos\vartheta \end{bmatrix}$	$\mathbf{t} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$
Uniform scaling by s	$\mathbf{A} = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix}$	$\mathbf{t} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$
Horizontal shear by m_x	$\mathbf{A} = \begin{bmatrix} 1 & m_x \\ 0 & 1 \end{bmatrix}$	$\mathbf{t} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$
Reflection against OX	$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$\mathbf{t} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Table 2.2. Exemplary fusion functions and some properties they fulfill: M – componentwise monotonicity, T – translation, uS – uniform scale, dS – d-scale, O – orthogonal, and A – affine equivariance.

function	**M**	**T**	**uS**	d**S**	**O**	**A**
CwAMean	●	●	●	●	●	●
CwMedian	●	●	●	●	○	○
1center$_{\mathfrak{d}_2}$	○	●	●	○	●	○
1median$_{\mathfrak{d}_2}$	○	●	●	○	●	○
TkMedian	○	●	●	●	●	●

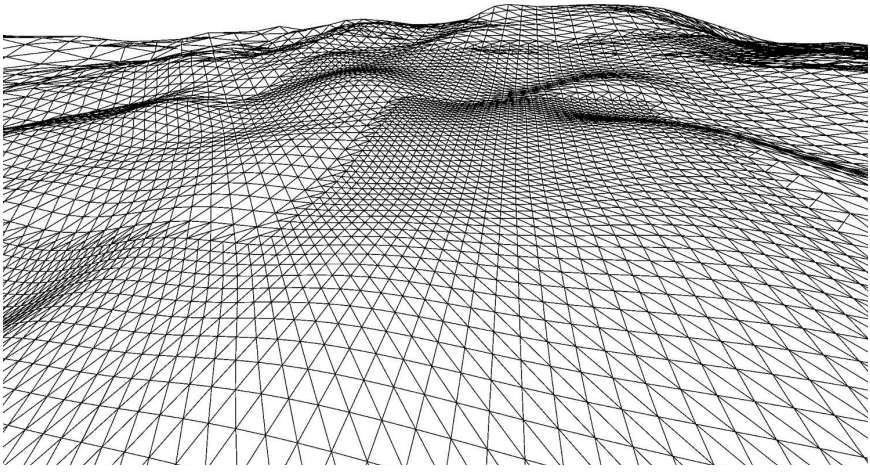


Figure 2.4. An exemplary virtual 3D world simulation.

Affinitization. It turns out that each fusion function F may be easily modified so that it fulfills affine equivariance. This may be done via the *transformation-retransformation* technique, see [113], Let us fix a set of d unique indices, $\mathcal{I} = \{i_0, i_1, i_2, \ldots, i_d\} \subseteq [n]$ and take:

$$\mathbf{A}_\mathcal{I} = [\mathbf{x}^{(i_1)} - \mathbf{x}^{(i_0)} \ \ldots \ \mathbf{x}^{(i_d)} - \mathbf{x}^{(i_0)}] \in \mathbb{R}^{d \times d}. \tag{2.14}$$

Assuming that the $\mathbf{A}_\mathcal{I}$ matrix is invertible, it can be treated as the basis matrix for a *data-driven coordinate system* in which a transformed version of our input data set is:

$$\mathbf{Y} = \mathbf{A}_\mathcal{I}^{-1} \mathbf{X}.$$

Thus, a modified version of a componentwise extension of $\mathsf{G} : \mathbb{R}^n \to \mathbb{R}$, AffG, may be given by:

$$\mathsf{AffG}(\mathbf{X}) = \mathbf{A}_\mathcal{I} \, \mathsf{CwG}\left(\mathbf{A}_\mathcal{I}^{-1} \left(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X})\right)\right) + \mathsf{CwAMean}(\mathbf{X}). \tag{2.15}$$

Such a construct is general, and its special case for $\mathsf{G} = \mathsf{Median}$ was first proposed by Chakraborty and Chaudhuri [109] (see also [369] for a discussion on affinitization of the 1-median). This time, unfortunately, the resulting fusion function is no longer symmetric.

Example 2.27. As a summary, Table 2.2 lists the properties fulfilled by idempotent fusion functions discussed so far. We see that the componentwise mean meets all of them.

2.3 Idempotence, internality, and weak monotonicity

First of all, let us note that if G is idempotent, then its componentwise extension CwG is also idempotent in the sense that for any $\mathbf{x} \in \mathbb{R}^d$ we have $\mathsf{CwG}(n * \mathbf{x}) = \mathbf{x}$.

More generally, if G is internal (recall Proposition 1.27), then $\mathsf{CwG}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$ is in the (axis-aligned) bounding box (orthotope, hyperrectangle) of $\mathbf{X}$:

$$\left[\bigwedge_{i=1}^n x_1^{(i)}, \bigvee_{i=1}^n x_1^{(i)}\right] \times \cdots \times \left[\bigwedge_{i=1}^n x_d^{(i)}, \bigvee_{i=1}^n x_d^{(i)}\right].$$

Remark 2.28. The above property is not necessarily an attractive generalization of ordinary internality: it seems to be too weak. Let $d = 2$, $n = 3$ and consider $\mathbf{x}^{(1)} = (1, 0)$, $\mathbf{x}^{(2)} = (0, 0)$, $\mathbf{x}^{(3)} = (0, 1)$.

For instance, if $\mathsf{G}(\mathbf{y}) = \frac{1}{n}\sum_{i=1}^n y_i$, then $\mathsf{CwG}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}) = (\frac{1}{3}, \frac{1}{3})$. However, if $\mathsf{G}(\mathbf{y}) = \bigvee_{i=1}^n y_i$, then $\mathsf{CwG}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}) = (1, 1)$.

In multivariate data analysis and computational geometry, the use of convex hulls is quite natural, see [324]. To recall, the convex hull $\mathrm{CH}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$ of a finite set of points is the smallest convex set (polytope) that includes all

the provided points. Equivalently, it is the set of all convex combinations of $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$:

$$\mathrm{CH}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \Big\{ \sum_{i=1}^{n} w_i \mathbf{x}^{(i)} : \text{for all vectors } \mathbf{w} \geq \mathbf{0} \text{ with } \sum_{i=1}^{n} w_i = 1 \Big\}.$$

The convex hull may be determined algorithmically. For example, if $d \in \{2, 3\}$, the Chan algorithm [110] has $O(n \log h)$ time complexity, where h is the number of vertices of $\mathrm{CH}(\mathbf{X})$. On the other hand, if $d > 3$, then an $O(n^{\lfloor d/2 \rfloor})$ algorithm exists [115].

Let us now introduce a new type of internality.

Definition 2.29. A fusion function F is *CH-internal* if and only if for all $\mathbf{X}$ we have that $\mathsf{F}(\mathbf{X}) \in \mathrm{CH}(\mathbf{X})$.

Please note that for $d = 1$ we have $\mathrm{CH}(\mathbf{X}) = [\bigwedge_{i=1}^{n} x_1^{(i)}, \bigvee_{i=1}^{n} x_1^{(i)}]$, i.e., it is the smallest real interval containing all the input samples. Because of that, for univariate fusion functions, the CH-internality and ordinary internality coincide.

One may wonder about the relationship between the CH- and bounding box-based internality. Of course, each CH-internal function fulfills the straightforward extension of ordinary internality. However, e.g., CwMedian is bounding box- but not CH-internal. The following result states that these two notions are equivalent when rotation equivariant fusion functions are concerned.

Proposition 2.30. *Let* $\mathsf{F} : (\mathbb{R}^d)^n \to \mathbb{R}^d$ *be rotation equivariant and such that for any* $\mathbf{X}$ *it holds that* $\mathsf{F}(\mathbf{X})$ *is in the bounding box of* $\mathbf{X}$*. Then* $\mathsf{F}(\mathbf{X}) \in \mathrm{CH}(\mathbf{X})$.

A simple proof of this proposition is based on the fact that the convex hull is equivariant to rotations and that it is a subset of the bounding box. Moreover, the convex hull may be expressed as the intersection of appropriate halfspaces [179]. $\mathbf{X}$ may always be rotated so that any convex hull's face is aligned within the axes. Then the hyperplane that includes such a face coincides with the hyperplane including the bounding box's face.

As for monotonicity, we already noted that componentwise nondecreasingness is problematic. Instead, however, we may consider a straightforward componentwise extension of weak monotonicity, compare Definition 1.62.

Definition 2.31. A fusion function $\mathsf{F} : (\mathbb{R}^d)^n \to \mathbb{R}^d$ is *weakly monotone* whenever $\mathsf{F}(\mathbf{X} + \mathbf{t}) \geq_d \mathsf{F}(\mathbf{X})$ for any $\mathbf{t} \geq_d (d * 0)$ and $\mathbf{X} \in (\mathbb{R}^d)^n$.

Surely, every translation equivariant fusion function is weakly monotone, but the converse is not necessarily true.

Here is a "multidimensional" counterpart of Proposition 1.65.

Proposition 2.32. *Let* $\mathsf{F} : (\mathbb{R}^d)^k \to \mathbb{R}^d$ *for some* k, $\mathsf{G}_1, \ldots, \mathsf{G}_k : (\mathbb{R}^d)^n \to \mathbb{R}^d$, *and* $\mathsf{H} : (\mathbb{R}^d)^n \to \mathbb{R}^d$ *be given by* $\mathsf{H}(\mathbf{X}) = \mathsf{F}(\mathsf{G}_1(\mathbf{X}), \ldots, \mathsf{G}_k(\mathbf{X}))$ *for* $\mathbf{X} \in (\mathbb{R}^d)^n$.

— *If* $\mathsf{F}, \mathsf{G}_1, \ldots, \mathsf{G}_k$ *are idempotent (respectively, bounding box-internal, CH-internal, translation, uniform scale, d-scale, orthogonal, affine equivariant), then* H *is also idempotent (respectively, bounding box-internal, CH-internal, and so forth).*

— *If* F *is weakly monotone and* $\mathsf{G}_1, \ldots, \mathsf{G}_k$ *are translation equivariant, then* H *is weakly monotone.*

Note that a different form of monotonicity could also be defined by requiring that if $(\forall i \in [n])\ \mathbf{x}^{(i)} \leq_d \mathbf{y}^{(i)}$, then $\mathsf{F}(\mathbf{X}) \not\geq_d \mathsf{F}(\mathbf{Y})$. However, it is not even fulfilled by the Euclidean 1-median, compare Example 2.7.

2.4 Data depth, corresponding medians, and ordering of inputs

The purpose of the notion of data depth is to measure how "central" or "deep" a point $\mathbf{y}$ is with respect to a point cloud $\mathbf{X}$. It may be used, e.g., to visualize (mostly bivariate) data sets [324], compute statistical hypothesis tests [119, 319], design control charts, and even support decision making [415]. It has been studied extensively by data analysts and computer scientists.

What is crucial to us in this monograph is that with any depth notion, its corresponding multidimensional median may be defined, which may serve as a robust estimator of location, see [8, 413, 432] for some surveys on the topic. A depth-based median is a point of the maximal depth (or the center of gravity of a set of points of maximal depth, if there is no single point with such a property).

Let us assume that $\mathbf{X}$ is a d-dimensional data set in *regular position*, i.e., with no more than d points lying in a $(d-1)$-dimensional subspace. In particular, in the bivariate case, we have that no more than two observations are colinear. In the following paragraphs we review the most notable data depth notions (like Tukey's, Liu's, and Oja's) and their corresponding affine equivariant medians. Later on we shall note that the concept of data depth leads to orderings of the input points, which will enable us to define new, quite interesting fusion functions.

It is assumed that the depth of a point $\mathbf{y} \in \mathbb{R}^d$ relative to $\mathbf{X} \in (\mathbb{R}^d)^n$ is quantified via a bounded function $\mathsf{depth} : \mathbb{R}^d \times (\mathbb{R}^d)^n \to [0, b]$ for some b. Zuo and Serfling in [494] list some desirable properties that this notion should fulfill, namely, for any $\mathbf{X}$ and $\mathbf{y}$ they require:

— affine invariance[1]: for all $\mathbf{A} \in \mathbb{R}^{d \times d}$ of full rank and $\mathbf{t} \in \mathbb{R}^d$:

$$\mathsf{depth}(\mathbf{A}\mathbf{y} + \mathbf{t}; \mathbf{A}\mathbf{X} + \mathbf{t}) = \mathsf{depth}(\mathbf{y}; \mathbf{X}),$$

[1]Note that in this book we made a clear distinction between equivariance and invariance to specific transformations.

— monotonicity relative to the deepest point: if $\mathbf{y} = \sup_{\mathbf{y}} \mathsf{depth}(\mathbf{y}; \mathbf{X})$, then for all $\mathbf{z}$ and $\alpha \in [0, 1]$ it holds:

$$\mathsf{depth}(\mathbf{z}; \mathbf{X}) \leq \mathsf{depth}(\alpha\mathbf{y} + (1 - \alpha)\mathbf{z}; \mathbf{X}),$$

— vanishing at infinity: $\mathsf{depth}(\mathbf{y}; \mathbf{X}) \to 0$ as $\|\mathbf{y}\| \to \infty$.

Note that depth notions are often considered in a statistical environment, so other properties may additionally be of interest, e.g., maximality at center: $\sup_{\mathbf{y}} \mathsf{depth}(\mathbf{y}; \mathbf{X}) = \mathsf{depth}(\boldsymbol{\mu}; \mathbf{X})$ where $\boldsymbol{\mu}$ is the point of symmetry of the empirical distribution of $\mathbf{X}$ (if it exists), etc.

2.4.1 Tukey's halfplane location depth and median

In 1974, Tukey [451] introduced the concept of the *depth* of a value y with respect to a unidimensional set of points $\mathbf{x} = (x_1, \ldots, x_n)$. It is defined as the minimum number of data points from $\mathbf{x}$ on the left and on the right of y:

$$\mathsf{tdepth}_1(y; x_1, \ldots, x_n) = |\{i : x_i \leq y\}| \wedge |\{i : x_i \geq y\}|. \tag{2.16}$$

The Tukey depth is related to the observations' ranking. The sample minimum and maximum are the points of depth 1, the median is of depth $n/2$ (the "deepest" value), and the first and the third quartiles are of depth $n/4$. As noted in [160], one can define trimmed means by, say, averaging points of depth $\geq n/10$. This notion has been used to develop robust regression techniques, see, e.g., [408].

As a matter of fact, Tukey in the same paper [451] introduced a generalization of this idea too. *The halfplane location depth* of $\mathbf{y}$ relative to $\mathbf{X}$ is the smallest number of points in $\mathbf{X}$ contained in any closed halfhyperplane with boundary line through $\mathbf{y}$. In other words, see also [160]:

Definition 2.33. The *Tukey depth* of $\mathbf{y} \in \mathbb{R}^d$ relative to $\mathbf{X} \in (\mathbb{R}^d)^n$ is an integer such that:

$$\begin{aligned} \mathsf{tdepth}_d(\mathbf{y}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) &= \min_{\mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\| = 1} \mathsf{tdepth}_1\left(\mathbf{u}^T\mathbf{y}; \mathbf{u}^T\mathbf{X}\right) \\ &= \min_{\mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\| = 1} |\{i : \mathbf{u}^T\mathbf{x}^{(i)} \geq \mathbf{u}^T\mathbf{y}\}|. \end{aligned}$$

Remark 2.34. Multidimensional Tukey depth is defined via projection pursuit, see [253]. It results in applying all possible one-dimensional projections of the data set to a line and computing the univariate Tukey depth.

It is easily seen that a set of all points of depth $\geq \delta$ (a δ-depth contour), for any given $\delta > 0$, is either empty or is a convex polytope (e.g., polygon for $d = 2$).

Note that a point outside the convex hull of $\mathbf{X}$ is always of depth 0 [410]. On the other hand, for all $\mathbf{y}$ we have $\mathsf{tdepth}_d(\mathbf{y}; \mathbf{X}) \leq n$. In fact, we may be slightly more precise about the upper limit for $d = 2$.

Proposition 2.35. *If $\mathbf{X}$ is a bivariate data set in regular position, then the maximal Tukey depth, $\delta = \max_{\mathbf{y}' \in \mathbb{R}^d} \mathsf{tdepth}_d(\mathbf{y}'; \mathbf{X})$, fulfills:*

$$\left\lceil \frac{n}{3} \right\rceil \leq \delta \leq \left\lfloor \frac{n}{2} \right\rfloor.$$

The upper bound was proved by Rousseeuw and Ruts in [410] while the lower bound was given by Donoho and Gasko [160]. More generally, for any d, by [160], we have that $\left\lceil \frac{n}{d+1} \right\rceil \leq \delta \leq \left\lceil \frac{n}{2} \right\rceil$.

Note that the deepest point might not be uniquely defined. In order to overcome this issue, we may consider the following fusion function.

Definition 2.36. Let $\mathcal{R}$ be the deepest Tukey depth region with respect to given $\mathbf{X} \in \mathbb{R}^{d \times n}$, i.e., $\mathcal{R} = \left\{\mathbf{y} \in \mathbb{R}^d : \mathsf{tdepth}_d(\mathbf{y}; \mathbf{X}) = \delta\right\}$, where $\delta = \max_{\mathbf{y}' \in \mathbb{R}^d} \mathsf{tdepth}_d(\mathbf{y}'; \mathbf{X})$. The center of gravity of such a region:

$$\mathsf{TkMedian}(\mathbf{X}) = \frac{\int_{\mathbb{R}^d} x \mathbf{1}(x \in \mathcal{R})\, dx}{\int_{\mathbb{R}^d} \mathbf{1}(x \in \mathcal{R})\, dx}, \tag{2.17}$$

is called the *Tukey median* of $\mathbf{X}$.

For $d = 1$, the Tukey median generalizes the concept of a median. Thus, in higher dimensions this fusion function can be thought of as a multidimensional median.

Example 2.37. Figure 2.5 depicts an exemplary data set, the three Tukey depth contours, and the center of gravity of the deepest Tukey depth region, i.e., the Tukey median.

Remark 2.38. Let $(x_1, \ldots, x_m)$ and $(y_1, \ldots, y_m)$ be coordinates of a convex polygon in $\mathbb{R}^2$, ordered clockwise. Then its center of gravity, (C_x, C_y), is given by:

$$\begin{aligned} C_x &= \frac{\sum_{i=1}^{m} (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i)}{3 \sum_{i=1}^{m} (x_i y_{i+1} - x_{i+1} y_i)}, \\ C_y &= \frac{\sum_{i=1}^{m} (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i)}{3 \sum_{i=1}^{m} (x_i y_{i+1} - x_{i+1} y_i)}, \end{aligned}$$

where, for brevity of notation, $x_{m+1} = x_1$ and $y_{m+1} = y_1$. For $d > 2$, e.g., one may perform a Delaunay triangulation of a given convex polytope and calculate sums of appropriate integrals (for each simplex independently).

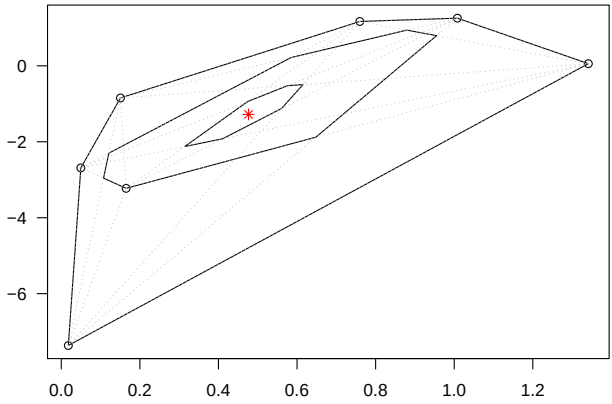


Figure 2.5. Tukey depth contours and Tukey median (∗) of a data set.

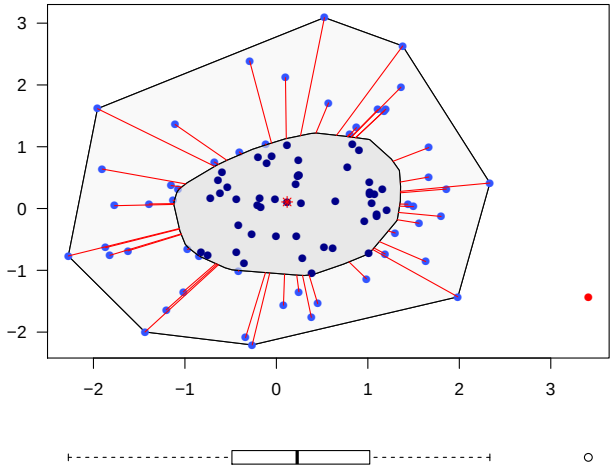


Figure 2.6. A bagplot of a bivariate data set and a boxplot of its projection onto OX generated with R (`aplpack::bagplot`).

Example 2.39. A *bagplot*, a bivariate version of the box-and-whisker plot, is based on the discussed notions, see Figure 2.6. It consists of the Tukey median, a *bag* that contains 50% of the data points (it results in a linear interpolation of two Tukey depth regions), and a *fence* that separates inliers form outliers (originally, an inflated version of the bag scaled by a factor of 3). For more details the reader is referred to [411].

Remark 2.40. The Tukey depth, as well as its corresponding median, is affine equivariant, see [160, Lemma 2.1]. Moreover, the Tukey depth is monotonic relative to the deepest point and vanishes at infinity.

For $d = 2$, a naïve algorithm to compute the Tukey depth requires $O(n^2)$ time. However, in [409] an optimal (see [9]) $O(n \log n)$ algorithm `LDEPTH` was given. It is implemented in R's `depth` package and available via a call to `depth(..., method="Tukey")`. For $d = 3$ there exists a $O(n^2 \log n)$ exact algorithm, see [412]. For larger d, there is an approximate Monte Carlo-type algorithm, also provided in [412].

Algorithm 2.41. *Here is how we may approximate* $\mathsf{tdepth}_d(\mathbf{y}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$ *for arbitrary d, see [412, Section 2.3].*

1. *Let* $D := n$*;*
2. *Repeat m times (for a given m):*
   2.1. *Draw a random sample of size d from* $\mathrm{U}\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$*;*
   2.2. *Determine a direction* $\mathbf{u}$ *perpendicular to the above subset;*
   2.3. *Project the points in* $\mathbf{X}$ *to the line L through* $\mathbf{y}$ *with direction* $\mathbf{u}$*;*
   2.4. *Compute the univariate Tukey depth* D' *of* $\mathbf{y}$ *on L;*
   2.5. *Set* $D := D \wedge D'$*;*
3. *Return D as result;*

A point with the largest Tukey depth (there may be many such points) may be found in expected $O(d \log n)$ time for $d = 2$ and in $O(n^{d-1})$ expected time for $d \geq 3$, see [111]. For $d = 2$ the ISODEPTH [417] algorithm determines vertices of a depth contour in $O(n^2 \log n)$ time and the HALFMED [410] algorithm for computing the Tukey median is $O(n^2 \log^2 n)$. Additionally, in [306] we may find an algorithm to compute a high depth point (not necessarily the median) in $O(n \log^2 n)$ and a lower bound for this task $\Omega(n \log n)$. The interested reader is referred to [8, 79] for further discussion on Tukey depth-related algorithms.

2.4.2 Liu's simplical depth and median

Note that for $i_1, \ldots, i_{d+1} \in [n]$, if $\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})}$ are affinely independent, then the convex hull of $d + 1$ points, $\mathrm{CH}\left(\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})}\right)$, defines a d-dimensional simplex. In particular, for $d = 2$, $\mathrm{CH}\left(\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_3)}\right)$ simply designates a triangle. Another notion of data depth of a point $\mathbf{y}$ relative to $\mathbf{X}$ is by Liu [323]. In the bivariate case it is defined as the number of triangles formed by any three elements in $\mathbf{X}$ that contain $\mathbf{y}$. Intuitively, a "deep" or "central" point is of large Liu depth. More generally, we have what follows.

Definition 2.42. The Liu *simplical depth* of $\mathbf{y} \in \mathbb{R}^d$ with respect to $\mathbf{X} \in \mathbb{R}^{d \times n}$ is defined as:

$$\mathsf{sdepth}(\mathbf{y}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \frac{\left|\left\{\{i_1, \ldots, i_{d+1}\} : \mathbf{y} \in \mathrm{CH}\left(\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})}\right)\right\}\right|}{\binom{n}{d+1}}.$$

It is easily seen that the Liu depth is affine invariant, that is its result does not change under arbitrary affine transformations.

Remark 2.43. Recall that it is easy to check whether $\mathbf{y} \in \mathrm{CH}(\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})})$. For a nondegenerate simplex, it suffices to solve $\mathbf{y} = [\mathbf{x}^{(i_1)} \ldots \mathbf{x}^{(i_{d+1})}]\,\boldsymbol{\alpha}$ for $\boldsymbol{\alpha}$ under the constraint $\sum_{i=1}^{d+1} \alpha_i = 1$ and verify whether $\boldsymbol{\alpha} \geq_{d+1} \mathbf{0}$.

The *simplical median*, SMedian, may be defined similarly to the Tukey median – as a point with the greatest simplical depth or the center of gravity of the deepest Liu depth region. This leads to an affine equivariant fusion function.

For $d = 2$, a straightforward algorithm to compute the simplical depth of a point requires $O(n^3)$ time. An optimal (see [9]) $O(n \log n)$ algorithm for that very purpose was proposed in [409]. It is available in R via a call to `depth::depth(..., method="Liu")`. Moreover, for $d = 2$ there exists an $O(n^4)$ time algorithm for finding the simplical median, see [10].

2.4.3 Oja's depth and median

The Oja depth (also known as the simplical volume depth) has been introduced in [378]. Here we provide its slightly transformed version, as given in [324], since the original definition is not compatible with the aforementioned depth measures: we would like to assure that a central point is of the greatest depth.

Definition 2.44. The *Oja depth* of $\mathbf{y} \in \mathbb{R}^n$ with respect to $\mathbf{X} \in (\mathbb{R}^d)^n$ is given by:

$$\mathsf{odepth}(\mathbf{y}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \binom{n}{d}^{-1} \frac{1}{1 + \sum_{\{i_1, \ldots, i_d\}} \mathrm{vol}\left(\mathrm{CH}\left(\mathbf{y}, \mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_d)}\right)\right)},$$

where $\mathrm{vol}(\cdot)$ designates the volume of a given simplex.

In particular, for $d = 2$, the Oja depth of a point is the sum of all the areas of triangles formed by this point and two points in an input data set.

Remark 2.45. The volume of a nondegenerate d-dimensional simplex given by vertices $\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})}$ equals to:

$$\operatorname{vol}\left(\operatorname{CH}\left(\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})}\right)\right) = \operatorname{abs}\left(\frac{1}{d!} \det \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1^{(i_1)} & x_1^{(i_2)} & \ldots & x_1^{(i_{d+1})} \\ \vdots & \vdots & \ddots & \vdots \\ x_d^{(i_1)} & x_d^{(i_2)} & \ldots & x_d^{(i_{d+1})} \end{bmatrix}\right).$$

This depth measure is not affine invariant. It is because for an affine transformation $T(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{t}$ we have:

$$\operatorname{vol}(T(\mathbf{x}^{(i_1)}), \ldots, T(\mathbf{x}^{(i_{d+1})})) = \operatorname{abs}(|\mathbf{A}|) \operatorname{vol}(\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})}),$$

see [378, Lemma 2.1]. However, its corresponding median, defined as a point with the maximum depth, is affine equivariant.

Definition 2.46. The *Oja median* of $\mathbf{X} \in \mathbb{R}^{d \times n}$ is given by:

$$\mathsf{OMedian}(\mathbf{X}) = \operatorname*{arg\,min}_{\mathbf{y} \in \mathbb{R}^d} \sum_{\{i_1, \ldots, i_d\}} \operatorname{vol}(\operatorname{CH}(\mathbf{y}, \mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_d)})). \tag{2.18}$$

Remark 2.47. Note that the Oja median generalizes the one-dimensional median, which is a point y that minimizes $\sum_i |x_i - y|$.

For $d = 2$ there exists an $O(n \log^3 n)$ algorithm [10] for finding the Oja median and an $O(n \log n)$ algorithm [11] for computing the Oja depth of a given point, see also `depth::depth(..., method="Oja")` in R. Another algorithm for finding the Oja median for any d was proposed in [406] and runs in $O(dn^d \log n)$ time.

Note that in [376] a generalization of the Oja median was proposed:

$$\mathsf{OMedian}_\alpha(\mathbf{X}) = \operatorname*{arg\,min}_{\mathbf{y} \in \mathbb{R}^d} \sum_{\{i_1, \ldots, i_d\}} \operatorname{vol}(\operatorname{CH}(\mathbf{y}, \mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_d)}))^\alpha \tag{2.19}$$

for some predefined $\alpha \in [1, 2]$.

2.4.4 Other depth notions

Below we list some other approaches for defining data depth.

L_1 depth. The L_1 depth by Vardi and Zhang [456] is closely connected to the 1-median given by Equation (2.7). In particular, L_1 depth is maximized at the 1-median.

Projection depth. The projection depth by Zuo [493] is a generalization of the concept by Donoho and Gasko [160], see also [413]. A measure of outlyingness of a point $\mathbf{y}$ is given via projection pursuit:

$$\mathsf{O}(\mathbf{y}; \mathbf{X}) = \sup_{\|\mathbf{u}\|=1} \frac{|\mathbf{u}^T\mathbf{y} - \mathsf{F}(\mathbf{u}^T\mathbf{y})|}{\mathsf{V}(\mathbf{u}^T\mathbf{y})}, \tag{2.20}$$

where F is some unidimensional aggregation function (e.g., median) and V is some spread measure (e.g., median absolute deviation, see Section 5.2). This leads to a depth measure:

$$\mathsf{pdepth}(\mathbf{y}; \mathbf{X}) = \frac{1}{1 + \mathsf{O}(\mathbf{y}; \mathbf{X})}. \tag{2.21}$$

Perihedral depth. Perihedral depth, see [174], is given by the number of subsets of $\mathbf{X}$ whose convex hulls contain $\mathbf{y}$.

Convex hull peeling depth. The convex hull peeling depth (see [178]; according to [252] the idea was proposed by Tukey) is determined by consecutively computing a convex hull of a set of points and removing values lying outside its boundary. The corresponding median may be constructed by computing the center of gravity of the "last" convex hull. According to [8], convex hull peeling may be done in $O(n \log^2 n)$ time for $d = 2$.

Delaunay depth. Recall that a Delaunay triangulation of $\mathbf{X}$ is a triangulation (tessellation) $\mathrm{DT}(\mathbf{X})$ such that all points in $\mathbf{X}$ are not in circum-hyperspheres of any simplices in $\mathrm{DT}(\mathbf{X})$. The Delaunay depth (according to [1] introduced by Green in [233]) of $\mathbf{y}$ with respect to $\mathbf{X}$ is the length of the shortest path in $\mathrm{DT}(\mathbf{X})$ from $\mathbf{y}$ to the convex hull of $\mathbf{X}$.

Example 2.48. Figure 2.7 depicts an exemplary Delaunay triangulation of a set of 5 points in $\mathbb{R}^2$. For each of the triangles in the triangulation, its circumcircle is also plotted.

In point of fact, the Delaunay depth is a member of a special class called *proximity depths*, defined as the number of edges in a proximity graph that must be visited to reach $\mathrm{CH}(\mathbf{X})$.

Zonoid data depth. The zonoid data depth, see [175], of $\mathbf{y} \in \mathbb{R}^d$ with respect to $\mathbf{X}$ is defined as:

$$\mathsf{zdepth}(\mathbf{y}; \mathbf{X}) = \sup\{\alpha \in [0, 1] : \mathbf{y} \in D_\alpha(\mathbf{X})\}, \tag{2.22}$$

with convention $\sup\{\emptyset\} = 0$, where D_α is the *α-trimmed region* [296] of the empirical distribution generated by $\mathbf{X}$, i.e.:

$$D_\alpha(\mathbf{X}) = \left\{ \sum_{i=1}^{n} w_i \mathbf{x}^{(i)} : \sum_{i=1}^{n} w_i = 1,\ (\forall i)\ w_i \geq 0,\ \alpha w_i \leq 1/n \right\}. \tag{2.23}$$

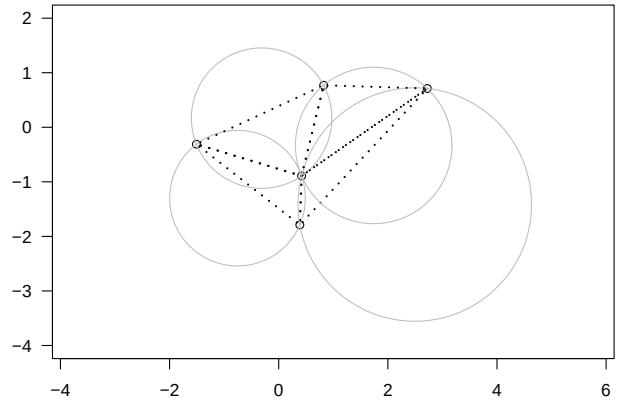


Figure 2.7. Delaunay triangulation of an exemplary bivariate set of points, together with circumcircles of all the triangles in the given tesselation.

Note that for $\alpha \in [0, 1/n]$ we have $D_\alpha = \mathrm{CH}(\mathbf{X})$. Moreover, D_1 is a singleton containing the centroid of $\mathbf{X}$ and for $\alpha < \alpha'$ we have $D_{\alpha'} \subset D_\alpha$. Of course, if $\mathbf{y} \notin \mathrm{CH}(\mathbf{X})$, then $\mathsf{zdepth}(\mathbf{y}; \mathbf{X}) = 0$. The zonoid data depth fulfills some important properties: it is affine invariant, continuous with respect to $\mathbf{y}$ and each $\mathbf{x}^{(i)}$, and monotone. The computation of depth of a given point may be reduced to a linear programming task, see [175]. However, the point with the greatest zonoid depth corresponds to $\mathsf{CwAMean}(\mathbf{X})$.

For a summary of other depth notions, such as the Mahalanobis depth, majority depth, or the likelihood depth, see [324].

Remark 2.49. Here is a possible application of the concept of data depth in regression analysis. Assume that we are given $\mathbf{X} \in (\mathbb{R}^d)^n$, $\mathbf{y} \in \mathbb{R}^n$ and we would like to fit a hyperplane $H_{\boldsymbol{\vartheta}}$, $\boldsymbol{\vartheta} \in \mathbb{R}^{d+1}$, defined by $y = \vartheta_1 x_1 + \cdots + \vartheta_d x_d + \vartheta_{d+1}$, such that $H_{\boldsymbol{\vartheta}}(\mathbf{X})$ is as close to $\mathbf{y}$ as possible.

The *regression depth* (introduced by Rousseeuw and Hubert in [408]) of $H_{\boldsymbol{\vartheta}}$ relative to $\mathbf{X}$ and $\mathbf{y}$ is defined as the smallest number of indices like i such that the residual $r_i = \vartheta_1 x_1^{(i)} + \cdots + \vartheta_d x_d^{(i)} + \vartheta_{d+1} - y_i$ needs to change its sign to make $H_{\boldsymbol{\vartheta}}$ *nonfit*, i.e., there exists a hyperplane V such that no $\mathbf{x}^{(i)}$ is on V, $r_i > 0$ for all $\mathbf{x}^{(i)}$ in one of V's open halfspace and $r_i < 0$ for all $\mathbf{x}^{(i)}$ in the other halfspace.

Intuitively, it is the smallest number of observations in $\mathbf{X}$ that would need to be removed in order to make a computed regression model a *nonfit*. It measures how well a hyperplane fit represents data: a good fit is of larger depth than a bad one. Thus, a fit with large depth is well-balanced relative to the input data.

There is an exact $O(n \log n)$-time algorithm for computing the Tukey-based regression depth for the case $d = 1$, see [408]. It was extended to arbitrary d in [412], but its time complexity is $O(n^d \log n)$; obviously, for large d and n such a routine is practically unusable. However, an approximate approach, similar to the one in Algorithm 2.41, may be used in such a case, see also [412, 414]. There is also an algorithm to compute hyperplanes with the greatest depths [455].

2.4.5 Symmetrization of fusion functions

Recall that a fusion function is symmetric, whenever for all permutations σ of $[n]$ it holds $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathsf{F}(\mathbf{x}^{\sigma(1)}, \ldots, \mathbf{x}^{\sigma(n)})$.

Given a non-symmetric unidimensional function, one may easily symmetrize it by referring to the notion of an order statistic, i.e., the ith smallest value among a set of input elements. It is because, by Proposition 1.80, $\mathsf{F} : \mathbb{R}^n \to \mathbb{R}$ is symmetric if and only if there exists a function $\mathsf{G} : \mathbb{R}^n \to \mathbb{R}$ such that:

$$\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathsf{G}(\mathbf{x}^{(\sigma(1))}, \ldots, \mathbf{x}^{(\sigma(n))}),$$

where σ is an ordering permutation of the input values. In such a way, e.g., a weighted arithmetic mean becomes an OWA operator. Such a construction is only valid, however, in the $d = 1$ case, as here a natural linear order $\leq$ is defined, see [324] for discussion.

In other words, if $d > 1$, then it is not easy to determine which values are "small" or "large", especially if we allow a set of points to be orthogonally transformed.

One possible way to order a set of points in $\mathbb{R}^2$ is to use one of the so-called *plane-filling curves*. For instance, let us consider the fractal-like Hilbert curve. Its building process is recursive and its first few steps are depicted in Figure 2.8. A set of points may be sorted by considering the order in which they appear on such a plane-filling curve. Notably, the CGAL [442] library has effective procedures to do so, also in higher dimensions. Such a sorting scheme may be used to speed up some geometric algorithms. Unfortunately, it is easily seen that the resulting ordering is neither translation nor, e.g., rotation invariant (but it might be made translation and uniform scale invariant by transforming the input data set).

Another way to sort a multivariate data set is to order the input values with respect to increasing distances from a fixed point, e.g., the set's componentwise mean. If the Euclidean distance is used, the introduced sorting scheme shall be affine equivariant. Yet, it might not be unique for some data sets. If ties occur, one may first order the observations relatively using the same ordering as in the input data set (this may be easily done by applying a stable sort algorithm).

More elaborate approaches may be based on the concept of data depth. With these, the points $\mathbf{x}^{(i)}$, $i = 1, \ldots, n$, may be ordered with respect to their decreasing or increasing depths. In other words, we may make use of a permutation σ of $\{1, \ldots, n\}$ such that $\sigma(i) \leq \sigma(j)$ implies that for $i < j$:

$$\mathsf{depth}(\mathbf{x}^{(\sigma(i))}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) \leq \mathsf{depth}(\mathbf{x}^{(\sigma(j))}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}),$$

where depth is some data depth measure. In this way, we get so-called *depth order statistics*, see [324]. Note that, unlike in the univariate case, they are not ordered from the "smallest" to the "largest", but from the "most central" to the "least central".

Having an ordered version of the input set of points, one may easily define, e.g., multidimensional versions of trimmed or Winsorized means, see [353].

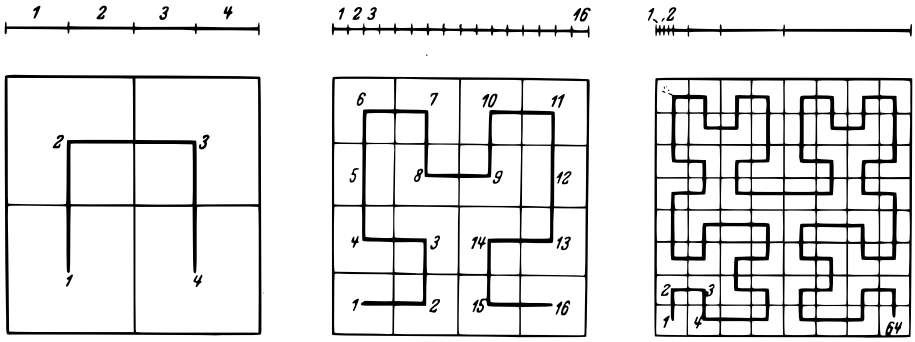


Figure 2.8. Three iterations of the Hilbert curve creation process as depicted in Hilbert's original 1891 paper [248].

2.5 Penalty-based fusion functions

At the very beginning of this chapter, we introduced some notable fusion functions: the componentwise mean, 1-median, and 1-center (with respect to the Euclidean metric). Let us now discuss them, as well as their generalizations, in greater detail.

2.5.1 1-median

Circa 1650, Evangelista Torricelli proposed a solution to a problem posed by Pierre de Fermat in the early 17th century: given three points in a plane, find the fourth point for which the sum of its distances to the three given points is as small as possible (compare [300]). This task can be formulated for an arbitrary number of points as follows. Find $\mathbf{y}$ such that:

$$\mathsf{1median}_{\mathfrak{d}}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \arg\min_{\mathbf{y} \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{y}), \tag{2.24}$$

where $\mathfrak{d}$ is a metric (originally the Euclidean one). Such a point, called in the literature the *1-median*, geometric median, spatial median, mediancenter, L_1-median, Fermat-Weber, or Torricelli point, generalizes the concept of a one-dimensional median (i.e., for $d = 1$ it is equal to Median for arbitrary L_p metric $\mathfrak{d}_p$ and odd n).

Euclidean metric. If $\mathfrak{d} = \mathfrak{d}_2$, the Euclidean 1-median is slightly less sensitive to outliers than the centroid (CwAmean, see below), compare Figure 2.9.

In the unidimensional case, as noted above, the solution reduces to the sample median and thus it might not be unique. However, for $d \geq 2$ and $\mathbf{X}$ such that it is not concentrated on a line, Milasevic and Ducharme showed [366] that the spatial median is always well-defined.

Note that Euclidean 1-median is sometimes used as an estimate of the underlying multidimensional probability distribution's theoretical median. Moreover,

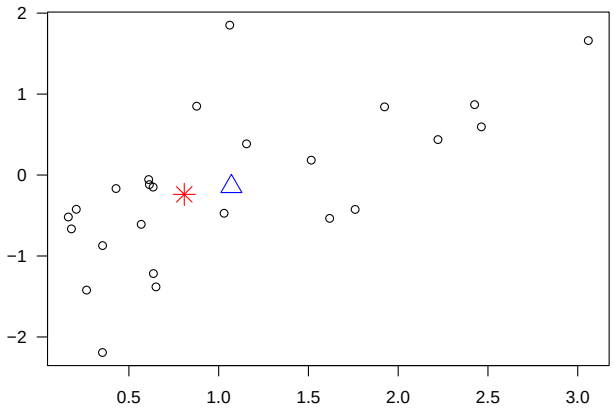


Figure 2.9. 1-median (∗) and centroid (△) of an exemplary data set. By definition, 1-median is less sensitive to outliers.

Brown in [86] generalized the two-sample statistical hypothesis sign test for the equality of medians in one dimension by using their spatial analogues (the angle test).

Weighted Euclidean metric. Let us consider a more general version of the above-presented case, closely related to the Fermat-Weber problem, see, e.g., [81, 441], which aims at finding the location for a new facility that minimizes the sum of transportation costs to n destination points (e.g., customers), having in mind different costs per unit distance.

Given a weighting vector $\mathbf{w}$, the *weighted geometric median* is defined as:

$$\mathsf{1median}_{\mathfrak{d}_2,\mathbf{w}}(\mathbf{X}) = \arg\min_{\mathbf{y}\in\mathbb{R}^d} \sum_{i=1}^{n} w_i \mathfrak{d}_2(\mathbf{x}^{(i)}, \mathbf{y}), \tag{2.25}$$

Unfortunately, in general, no analytic formula expressing the solution to the above equation exists, even in the $(\forall i)\ w_i = 1/n$ case. By considering the partial derivatives of the above objective function, it may be shown, see [456], that it is a point $\mathbf{y}$ such that:

$$\sum_{i=1}^{n} \frac{w_i \mathbf{y}}{\mathfrak{d}_2(\mathbf{x}^{(i)}, \mathbf{y})} = \sum_{i=1}^{n} \frac{w_i \mathbf{x}^{(i)}}{\mathfrak{d}_2(\mathbf{x}^{(i)}, \mathbf{y})}. \tag{2.26}$$

However, from Equation (2.26), we may derive the following algorithm to compute the fusion function of interest.

Algorithm 2.50. *Weiszfeld procedure [465]:*

1. *Choose a starting point* $\mathbf{y}^{(0)}$ *in the convex hull of* $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$*;*
2. *For* $j = 1, 2, \ldots$ *do:*

2.1. If $\mathbf{y}^{(j-1)} = \mathbf{x}^{(i)}$ *for some* $i \in [n]$*, then let* $\mathbf{y}^{(j)} := \mathbf{x}^{(i)}$*;*

2.2. Otherwise, let $\mathbf{y}^{(j)} := \dfrac{\sum_{i=1}^{n} \dfrac{w_i \mathbf{x}^{(i)}}{\mathfrak{d}_2(\mathbf{x}^{(i)} - \mathbf{y}^{(j-1)})}}{\sum_{i=1}^{n} \dfrac{w_i}{\mathfrak{d}_2(\mathbf{x}^{(i)} - \mathbf{y}^{(j-1)})}}$*;*

2.3. If $\mathfrak{d}_2(\mathbf{y}^{(j)}, \mathbf{y}^{(j-1)}) \leq \varepsilon$ *for some fixed* $\varepsilon > 0$*, then return* $\mathbf{y}^{(i)}$ *as result.*

It may be shown, see [81], that the Weiszfeld algorithm converges to an optimal solution for all but a countable set of starting points $\mathbf{y}^{(0)}$. An exemplary implementation of the above algorithm is given in Figure A.12, see also its more robust version called SOR-Weiszfeld introduced in [268] and the AS78 algorithm [228] which is based on the steepest descent heuristic.

Note that Equation (2.26) implies that:

$$\mathbf{y} = \sum_{i=1}^{n} v_i \mathbf{x}^{(i)}, \quad v_i = \frac{\dfrac{w_i}{\mathfrak{d}_2(\mathbf{x}^{(i)}, \mathbf{y})}}{\sum_{i=1}^{n} \dfrac{w_i}{\mathfrak{d}_2(\mathbf{x}^{(i)}, \mathbf{y})}}. \tag{2.27}$$

We see that $(v_1, \ldots v_n)$ is a weighting vector. Hence, the 1-median fulfills the convex hull-based internality. Moreover, it is orthogonal, uniform scale, and translation equivariant but not d-scale and thus not affine equivariant, see [369] for discussion. Also, its symmetry depends solely on the form of the weighting vector $\mathbf{w}$.

Manhattan distance. Interestingly, it turns out that by setting $\mathfrak{d}$ to be the Manhattan $\mathfrak{d}_1$ metric, we get the already mentioned componentwise median, CwMedian, see [29]. Recall that this fusion function is nondecreasing, translation and d-scale equivariant, but not rotation equivariant (note how the Manhattan distance behaves under rotations). Note that the k-medians algorithm (more robust to outliers than k-means) was originally based on the 1-median with respect to $\mathfrak{d}_1$.

Other Minkowski distances. There exists a Newton-Raphson-like algorithm [29] which computes the 1-median in the case $\mathfrak{d}_p$ for arbitrary $p \geq 1$. As a matter of fact, for moderate values of p and sample sizes, this task may be easily determined using a generic nonlinear optimization solver, for example:

```
one_median_Lp <- function(X, p) {
   optim(rowMeans(X), function(c) {
      sum(colSums(abs(X-c)^p)^(1/p))
   }, method="BFGS", control=list(reltol=1e-16))$par
}
```

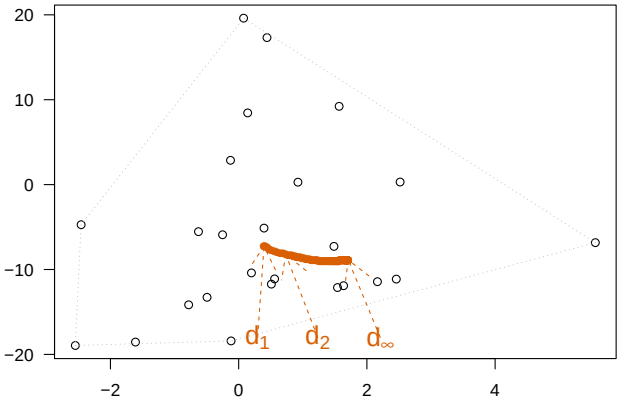


Figure 2.10. $\mathsf{1median}_{\mathfrak{d}_p}$ trace as a function of $p \in [0, \infty]$.

Example 2.51. Figure 2.10 depicts 1-median trace of an exemplary data set. It is assumed that the 1-median is computed with respect to Minkowski $\mathfrak{d}_p$ metrics and the trace is generated by varying $p \in [1, \infty]$.

2.5.2 Medoid

The 1-median should not be confused with the concept of a *medoid* or set median, which is a point $\mathbf{y}$ such that:

$$\mathsf{Medoid}_{\mathfrak{d}}(\mathbf{y}) = \operatorname*{arg\,min}_{\mathbf{y} \in \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}} \frac{1}{n} \sum_{i=1}^{n} \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{y}), \tag{2.28}$$

for arbitrary metric $\mathfrak{d}$ (usually Euclidean or Manhattan). The difference is that we do not look among all the vectors in $\mathbb{R}^d$, but restrict ourselves to the input data set (hence, the medoid is a kind of *exemplar*, compare one of possible definitions of internality on page 106). In other words, a medoid is a point in a given data set, for which average dissimilarity to all the other objects in the set is minimal.

Remark 2.52. A medoid may be non-uniquely defined. This is the case for $\mathfrak{d}_2$ and three vertices of an equilateral triangle (or more generally, d vertices of a regular d simplex). In such a situation, the computer science perspective is to choose *any* point that fulfills Equation (2.28). Yet, for $\mathsf{Medoid}_{\mathfrak{d}}$ to be a proper fusion function, we should choose some method of distinguishing *the* medoid of interest. In particular, we may assume that we return one that has the smallest index i among $\{i \in [n] : \mathbf{x}^{(i)} = \mathbf{y}\}$.

Medoids are useful, e.g., in clustering problems (the k-medoids algorithm, see [384]) or as rough estimates of 1-medians. They are internal as well as translation, uniform scale, and rotation equivariant for $\mathfrak{d} = \mathfrak{d}_2$.

Note that we shall refer to this concept once again when discussing aggregation in arbitrary pseudometric spaces, see Section 4.6.

2.5.3 Centroid

Given a weighting vector $\mathbf{w}$, the *weighted centroid* is a point $\mathbf{y}$ such that:

$$\mathbf{y} = \operatorname*{arg\,min}_{\mathbf{y}\in\mathbb{R}^d} \sqrt{\sum_{i=1}^{n} w_i \left(\mathfrak{d}_2(\mathbf{x}^{(i)}, \mathbf{y})\right)^2}, \tag{2.29}$$

where $\mathfrak{d}_2$ is the Euclidean metric.

Please notice the similarity between the above definition and the definition of the weighted Euclidean 1-median. ${\mathfrak{d}_2}^2$ is of course no longer a metric, but a kind of *dissimilarity measure*. Due to this simplification it turns out that the solution to the above equation is very easy: it is the componentwise extension of the weighted arithmetic mean. Thus, it is componentwise monotonic. Moreover, we already noted that it is an affine invariant fusion function which fulfills convex hull-based internality. Also note that the centroid minimizes the variance of distances from the observations to itself.

The centroid is a basis for the k-means clustering algorithm, see [197, 331]. On the other hand, its weighted version is used in the fuzzy c-means procedure [55]. In physics, the discussed notion reflects the center of mass of a system of particles.

Notably, the centroid is a special case of the Fréchet mean for $\mathfrak{d} = \mathfrak{d}_2$.

2.5.4 1-center

For a given metric $\mathfrak{d}$ the *1-center* (*smallest enclosing ball*, seb) problem aims at finding:

$$\mathsf{1center}_{\mathfrak{d}}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \operatorname*{arg\,min}_{\mathbf{y}\in\mathbb{R}^d} \bigvee_{i\in[n]} \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{y}). \tag{2.30}$$

In particular, if $\mathfrak{d}$ is the Euclidean metric $\mathfrak{d}_2$, the above task is called the Euclidean 1-center problem and was first proposed by James Sylvester in 1857 [438]. Note that this task is not the same as finding the center of a circumscribed circle.

Figure 2.11 depicts Euclidean 1-centers of two exemplary two-dimensional data sets. Such a formulation is used in many real-world applications, see, e.g., [219], which include: pattern recognition (finding reference points), computational biology (protein analysis), support vector machines – high-dimensional clustering, and nearest neighbor search. In particular, for $d = 3$ these may be used in computer graphics, e.g., visibility culling, ray tracing, and object collision detection. However, we should be careful when using it in data analysis: it is extremely sensitive to outliers. What is more, for $d = 2$ we have an important operational research application, known as the facility location problem, when one aims to seek the location of the distribution center that minimizes the distance to a customer that is situated farthest away.

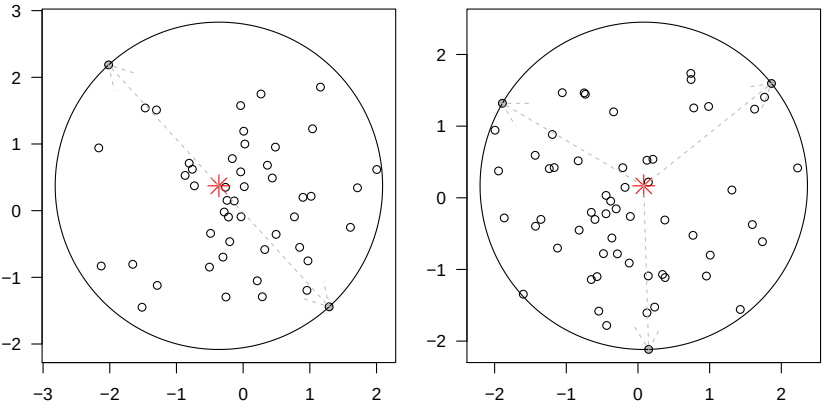


Figure 2.11. Euclidean 1-centers of two exemplary data sets.

It may be shown, see [220], that the solution to the Euclidean 1-center problem can be expressed as:

$$\mathbf{y} = \sum_{i=1}^{n} v_i \mathbf{x}^{(i)} = \mathbf{v}\mathbf{X}^T, \tag{2.31}$$

where the weighting vector $\mathbf{v}$ is computed by solving the quadratic programming (QP) problem:

$$\text{minimize } \mathbf{v}^T\mathbf{X}^T\mathbf{X}\mathbf{v} - (\text{diag}(\mathbf{X}^T\mathbf{X}))^T\mathbf{v} \quad \text{w.r.t. } \mathbf{v}$$

subject to:

$$\begin{aligned} \mathbf{1}^T\mathbf{v} &= 1, \\ \mathbf{v} &\geq_n \mathbf{0}. \end{aligned}$$

Note again that special care should be taken while choosing a software library to compute this QP task, compare Remark 1.160. For example, the quadprog package for R, which implements the dual method of Goldfarb and Idnani [227], is only able to find a solution if $\mathbf{D}$ is positive-definite, which – in general – is not our case. Instead, for this task we may use a generic QP solver given in Figures A.2 and A.3, which relies on the CGAL library. Figure A.10 gives an exemplary Rcpp implementation of a routine to compute the smallest enclosing ball.

A different, combinatorial algorithm (that resembles the simplex algorithm for linear programming) has been proposed in [187]. Moreover, the CGAL [442] library includes an implementation of Welzl's routine [466].

From Equation (2.31) it follows that the Euclidean 1-center is necessarily convex hull internal. What is more, it is translation, orthogonal, and uniform scale equivariant (but not d-scale equivariant).

On the other hand, the Chebyshev 1-center is a componentwise extension of $\mathsf{F}(\mathbf{x}) = (\mathsf{Max}(\mathbf{x}) + \mathsf{Min}(\mathbf{x}))/2$, i.e., it is the center of the points' bounding rectangle.

Moreover, similarly to the concept of a medoid, we may define a *seboid*, which is an exemplar minimizing the function in Equation (2.30), see Section 4.6 for further information.

2.5.5 A more general framework

Similarly as in Definition 1.105, we may introduce the notion of a penalty-based fusion function for aggregation of points in $\mathbb{R}^d$. This time, however, we should rather assume that the set of minimizers of a penalty function $P : \mathbb{R}^d \times (\mathbb{R}^d)^n \to [0, \infty]$ is a convex polytope and that a P-based fusion function F is given as the center of gravity of such a set. For $d = 1$, this setting generalizes the one from the previous chapter. Surely, each idempotent function F is a penalty-based one for some P.

Let depth be some data depth notion which is bounded from above by a value $m \in [0, \infty]$. By setting $P(\mathbf{y}; \mathbf{X}) = m - \mathsf{depth}(\mathbf{y}; \mathbf{X})$ we get that the median corresponding to depth is a P-based fusion function.

All the other fusion functions presented in this section may be generalized as follows, see Table 2.3

Definition 2.53. Let $\mathsf{D} : [0, \infty]^n \to [0, \infty]$ be a nondecreasing fusion function such that $\mathsf{D}(n * 0) = 0$ and $\mathfrak{d}$ be an arbitrary pseudometric. Then a *distance*-based penalty function is given by:

$$P(\mathbf{y}; \mathbf{X}) = \mathsf{D}\left(\mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{y}), \ldots, \mathfrak{d}(\mathbf{x}^{(n)}, \mathbf{y})\right). \tag{2.32}$$

Note that not all metrics lead to proper penalty functions, though. This is the case of the Hamming distance (see below).

Proposition 2.54. *If P is a distance-based penalty function generated by D, $\mathfrak{d}$, and F is a P-based fusion function, then we observe the following regularities:*

- *F is idempotent.*
- *If $\mathfrak{d}$ is a norm-generated metric, then F is translation equivariant.*
- *If $\mathfrak{d}$ is a norm-generated metric and D is scale equivariant, then F is uniform scale equivariant.*
- *If $\mathfrak{d}$ is the Euclidean metric, then F is orthogonal equivariant.*
- *If $\mathfrak{d}$ is the Euclidean metric and D is strictly increasing, then F is CH-internal.*
- *If $\mathfrak{d}$ is the Manhattan metric and D is strictly increasing, then F is bounding box-internal.*

Table 2.3. Examples of distance-based penalty functions.

D	D **minimizer**
Arithmetic mean	1-median
Weighted arithmetic mean	Weighted 1-median
Maximum	1-center
Quadratic mean	Centroid
Weighted quadratic mean	Weighted centroid

Other fusion functions may be used instead of those listed in Table 2.3, for example $\mathsf{D}(\mathbf{d}) = d_{(\lfloor n/2 \rfloor)}$, will give us the center of the smallest ball containing approximately half of the input points (may be useful in the process of constructing metric tree-based data structures, e.g., vp-trees [487]).

Remark 2.55. The idea of incorporating generic penalty minimizers in clustering tasks was discussed by Leisch in [314]. He proposes a generalization of the k-means and k-medians algorithm which works for any metric and its minimizer. To recall, the aim of such algorithms is to find, for a given k, the centers of clusters $\boldsymbol{\mu}^{(1)}, \ldots, \boldsymbol{\mu}^{(k)}$ which partition input data points into k disjoint groups. The ith point's membership to one of the clusters, $c(i) \in [k]$, is expressed in terms of its proximity to one of the cluster centers (cluster centers generate Dirichlet (Voronoi) regions, see Figure 2.12, which determine a center's "attraction area"). This type of clustering algorithms tries to approach a solution such that the total distance between all input points $\mathbf{x}^{(i)}$ and their corresponding clusters' centers $\boldsymbol{\mu}^{(c(i))}$ is as small as possible, i.e.:

$$\text{minimize} \sum_{i \in n} \mathfrak{d}(\mathbf{x}^{(i)}, \boldsymbol{\mu}^{(c(i))}) \quad \text{w.r.t. } c : [n] \to [k] \text{ (onto)},$$

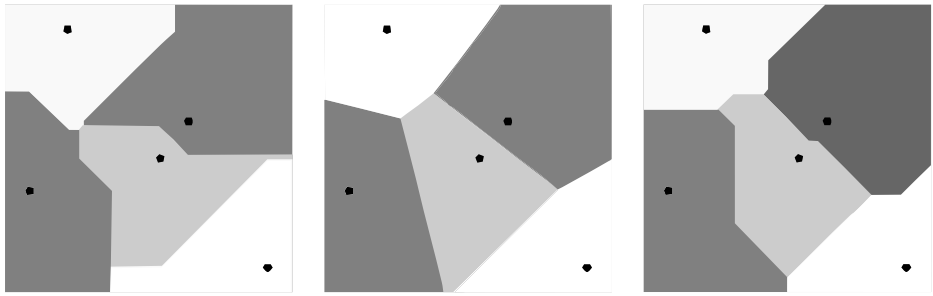
where:

$$\boldsymbol{\mu}^{(i)} = \operatorname*{arg\,min}_{\mathbf{y} \in \mathbb{R}^d} \mathsf{F}\{\mathbf{x}^{(j)} : j \in [n], c(j) = i\}. \tag{2.33}$$

A k-means-like algorithm is a heuristic which aims to solve the above optimization problem in the following manner:

1. Initialize $c(i), i \in [n]$, e.g., randomly;
2. Update the centroids according to Equation (2.33);
3. Repeat Step 2. until convergence.

Remark 2.56. There are various ways that can aid in choosing a fusion function for practical use. One of them may be based on the set of useful properties (such as a particular type of equivariance) that an aggregation method fulfills.



(a) $\mathfrak{d}_1$ (b) $\mathfrak{d}_2$ (c) $\mathfrak{d}_\infty$

Figure 2.12. Dirichlet (Voronoi) regions generated by 5 points in $\mathbb{R}^2$ and different metrics.

The other ones rely on a fusion function's numerical characteristics (compare the notion of a breakdown point in Section 5.5).

Nevertheless, these properties or characteristics are non-probabilistic in their nature. As multidimensional fusion functions are frequently investigated by computational statisticians, it is also interesting to inspect their behavior on random input data.

For instance, Massé and Plante in [354] perform a Monte Carlo study on the accuracy and robustness of ten bivariate location estimators: the centroid, Tukey Median, Liu median, Oja median, depth-based trimmed medians (Liu and Tukey, $\alpha = 0.05, 0, 1$), spatial median, and componentwise median. They consider 26 random data scenarios for different n and $d = 2$, including various types of samples' contamination, with the point of reference set to the population median (center of symmetry). It turns out that the best performance is exhibited by the Euclidean 1-median, the Oja and the Tukey median, as well as the componentwise median.

2.6 Aggregation on product lattices

In Section 1.7 we explored the topic of fusion of data which were objects in some bounded poset. Let us extend the discussion slightly to the case of information items that are instances of poset sequences. This is exactly the situation, for example, occurring in a decision making task where n experts express their opinions on d alternatives and there is a need to obtain their "averaged" view on all of the alternatives.

2.6.1 Cartesian product

The Cartesian product (see, e.g., [62]) of d identical bounded posets $\mathcal{P} = (P, \sqsubseteq, \underline{0}, \overline{1})$ is the bounded poset $\mathcal{P}^d = (P^d, \sqsubseteq^d, \underline{0}^d, \overline{1}^d)$ with $P^d = P \times \cdots \times P$, $\underline{0}^d =$

$(d * \underline{0})$, $\overline{1}^d = (d * \overline{1})$. Here, the partial ordering relation $\sqsubseteq^d$ is given by:

$$(x_1, \ldots, x_d) \sqsubseteq^d (y_1, \ldots, y_d) \iff x_1 \sqsubseteq y_1 \text{ and } \ldots \text{ and } x_d \sqsubseteq y_d. \tag{2.34}$$

Remark 2.57. Most constructions presented in this section may be quite easily extended to the case of a Cartesian product of non-identical bounded posets. We do not follow such a route for better readability of the material.

Additionally, if we consider a product of d identical bounded lattices $(P, \sqsubseteq, \sqcap, \sqcup, \underline{0}, \overline{1})$, then we get a bounded lattice with join $\sqcap^d$ and meet $\sqcup^d$ operations, respectively, given by:

$$\begin{aligned}(x_1, \ldots, x_d) \sqcap^d (y_1, \ldots, y_d) &= (x_1 \sqcap y_1, \ldots, x_d \sqcap y_d),\\ (x_1, \ldots, x_d) \sqcup^d (y_1, \ldots, y_d) &= (x_1 \sqcup y_1, \ldots, x_d \sqcup y_d).\end{aligned}$$

Remark 2.58. If $\mathcal{P}$ is a bounded chain, then $\mathcal{P}^d$ is a bounded lattice (product of chains is only a chain in trivial cases: $d = 1$ or $|P| = 1$). For instance, let $\mathcal{P} = ([0,1], \leq, \wedge, \vee, 0, 1)$. Considering $\mathcal{P}^2$, we have $(0,1) \not\leq_2 \not\geq_2 (1,0)$, hence $\leq_2$ is not a linear order.

Similarly, any function $\mathsf{F} : P^n \to P$, i.e., taking n objects in P as input, may be extended in a componentwise manner. This way, we obtain $\mathsf{F} : (P^d)^n \to P^d$ as follows:

$$\mathsf{F}^d(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \Big(\mathsf{F}(x_1^{(1)}, \ldots, x_1^{(n)}), \ldots, \mathsf{F}(x_d^{(1)}, \ldots, x_d^{(n)})\Big) \tag{2.35}$$

for all $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in P^d$.

Recalling the discussion on componentwise fusion functions in the case $P = \mathbb{R}$, we have the following result.

Proposition 2.59. *If F is an aggregation function on a bounded poset P, see Definition 1.180, then F^d is an aggregation function on P^d. More generally, if $\mathsf{F}_1, \ldots, \mathsf{F}_d$ are aggregation functions on P, then the componentwise (decomposable, see [293]) fusion function $(\mathsf{F}_1, \ldots, \mathsf{F}_d)$ is an aggregation function on P^d.*

As we know from previous sections, of course, one does not have to limit him/herself to such simple extensions of fusion functions. If some kind of dependency between variables exists in an input data set, more elaborate solutions may be necessary. For instance, in decision making we may want to introduce fusion functions that ignore the answers of experts who constantly (for all the attributes) provide contrasting answers. We may also do so for experts whose answers are characterized by a very small variability, and so forth.

2.6.2 Penalty-based aggregation on product lattices

Assume that $P = \{p_1, p_2, \dots\}$ is countable and that $\sqsubseteq$ is a linear order with $p_i \sqsubseteq p_j$ whenever $i \leq j$. We may consider a *natural metric* on P^d (see [91]) such that for any $(p_{i_1}, \dots, p_{i_d}), (p_{j_1}, \dots, p_{j_d}) \in P^d$ it holds:

$$\mathfrak{d}_{\mathrm{N}}\left((p_{i_1}, \dots, p_{i_d}), (p_{j_1}, \dots, p_{j_d})\right) = \sum_{u=1}^{d} |i_u - j_u|. \tag{2.36}$$

With this metric, penalty-based fusion functions such as some of those considered in Section 2.5 may be introduced. Note that a form of weighting of different dimensions and relative elements' order may also be incorporated here so that we get:

$$\mathfrak{d}_{\mathbf{w},\boldsymbol{\varphi}}\left((p_{i_1}, \dots, p_{i_d}), (p_{j_1}, \dots, p_{j_d})\right) = \sum_{u=1}^{d} w_u |\varphi_u(i_u) - \varphi_u(j_u)|. \tag{2.37}$$

for some increasing and convex $\varphi_1, \dots, \varphi_d : \mathbb{N} \to \mathbb{R}$ and a weighting vector $\mathbf{w} \in [0,1]^d$.

2.6.3 Conjunctive, disjunctive, and averaging functions

Let us go back to the Komorníková-Mesiar classification of fusion functions, see [293] and Section 1.7.3.

De Baets and Mesiar in [138] showed that the componentwise extension of d t-norms is also a t-norm. On the other hand, as shown by Jenei and De Baets in [261], there may exist t-norms on product lattices P^d that are not direct products of t-norms on P.

By [293, Proposition 5], we have that $(\mathsf{F}_1, \dots, \mathsf{F}_d)$ is strongly conjunctive (disjunctive) if and only if each F_i is strongly conjunctive (disjunctive).

Other properties are not necessarily inherited so easily as indicated in the following example.

Example 2.60 ([293]). Consider the bounded chain $([0,1], \leq, 0, 1)$. Here, the sample median Median is strongly averaging. But if we act on a product of three such chains we get that Median^3 is not even weakly averaging.

2.6.4 Other orders on product lattices

It turns out that the product order is only one of many possible extensions of $\sqsubseteq$ to a product lattice. Other popular choices include the inf-, sup-based, and lexicographic ordering. In decision making these correspond to maximin, maximax (see, e.g., [161]) and leximin (see, e.g., [162, 189]) approaches, respectively.

Definition 2.61. Let $(P, \sqsubseteq, \sqcap, \sqcup)$ be a lattice. Then the *inf-based ordering* is given for every $\mathbf{p}, \mathbf{q} \in P^d$ by $\mathbf{p} \sqsubseteq_{\min} \mathbf{q}$ if and only if $\bigsqcap_{i=1}^{d} p_i \sqsubseteq \bigsqcap_{i=1}^{d} q_i$.

Definition 2.62. The *sup-based ordering* is given for every $\mathbf{p}, \mathbf{q} \in P^d$ by $\mathbf{p} \sqsubseteq_{\max} \mathbf{q}$ if and only if $\bigsqcup_{i=1}^{d} p_i \sqsubseteq \bigsqcup_{i=1}^{d} q_i$.

In other words, the two above orders say that a lattice element $\mathbf{p}$ is dominated by $\mathbf{q}$, whenever the satisfaction degree of the least (respectively, greatest) satisfied constraint in the first object is not greater than the corresponding observation in the second one.

Definition 2.63. The *lexicographic ordering* is given for every $\mathbf{p}, \mathbf{q} \in P^d$ by:

$$\mathbf{p} \sqsubseteq_{\text{lex}} \mathbf{q} \iff (\exists i \in [d])(\forall j \in [i-1])\ p_j = q_j \text{ and } p_i \sqsubset q_i \text{ if } i < d \text{ and } p_i \sqsubseteq q_i \text{ otherwise.} \tag{2.38}$$

Note that an extended version of the above order shall be studied in the next Chapter.

Remark 2.64. As we already mentioned, the lexicographic order is particularly appealing in decision making. Imagine we have a set of criteria, ordered with respect to their importance, like "child safety", "price", and "attractive outlook" in the case of a decision making task to determine which car should be bought by an agent. If a car A is less safe than B, no matter what the satisfaction degrees of other criteria are, B is preferred to A. On the other hand, if A is as safe as B, then one should also consider its price and then – perhaps – its general appearance.

In the three discussed cases if $\sqsubseteq$ is a linear order, then the above-defined orders are at least total preorders.

Note that some of the results presented in Section 1.7 may be utilized in any of these new, "multidimensional" settings. A combination of posets gives us yet another poset and the methods presented in the previous Chapter are still valid here. This is because they are very general in their nature. We therefore decide not to explore them any further in this book.

2.7 Aggregation of character sequences

In Section 1.8 we noted that aggregation of n elements on a nominal scale was neither very challenging nor interesting. Nevertheless, the situation is quite different in the case of n vectors of length d with elements in some alphabet

Σ (and will be even more engaging in the next chapter, where we deal with character strings).

All the fusion functions considered in this section are distance penalty-based ones. Perhaps the most frequently used metric on Σ^d is the one introduced by Hamming, see [239].

Definition 2.65. The *Hamming distance* is defined for $\mathbf{x}, \mathbf{y} \in \Sigma^d$ as:

$$\mathfrak{d}_{\mathrm{H}}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} \mathbf{1}(x_i \neq y_i). \tag{2.39}$$

In other words, it is the total number of indices at which two given vectors differ.

Exemplary applications of fusion functions based on the Hamming distance minimizers include finding gene clusters, creating diagnostic probes, or discovering potential drug targets, see, e.g., [305], especially if we compute them over the DNA or protein sequences domain. Also, they are useful in error correction tasks: imagine that a few signals were sent with errors, the "central" one (this is particularly the case of the median vector discussed below) may represent the underlying correct information piece.

Let us briefly review possibly interesting properties of such fusion functions. Of course, there is no ordering relation on Σ, thus we cannot refer to any notion of monotonicity here. Instead, we may consider if for every $\mathbf{x}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in \Sigma^d$ it holds:

— $\mathsf{F}(n * \mathbf{x}) = \mathbf{x}$, (idempotency)

— $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathsf{F}(\mathbf{x}^{(\sigma(1))}, \ldots, \mathbf{x}^{(\sigma(n))})$ for all $\sigma \in \mathfrak{S}_{[n]}$, (symmetry)

— if $\mathbf{y} = \mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$, then $y_i \in \{x_i^{(1)}, \ldots, x_i^{(n)}\}$, (internality)

— $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathsf{F}(\mathbf{x}'^{(1)}, \ldots, \mathbf{x}'^{(n)}) = \mathbf{y}$ where $\mathbf{x}'^{(j)} \in \{\mathbf{x}^{(j)}, \mathbf{y}\}$, (decomposability)

and for extended fusion functions:

— $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathbf{y}$, then $\mathsf{F}(k * \mathbf{y}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathbf{y}$ for all k, (L-stability)

— $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathbf{y}$, then $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}, k * \mathbf{y}) = \mathbf{y}$ for all k. (R-stability)

2.7.1 Median

Let us first study the problem of finding:

$$\mathsf{Median}_{\mathfrak{d}_{\mathrm{H}}}(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)}) = \operatorname*{arg\,min}_{\mathbf{x}\in\Sigma^d} \sum_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)},\mathbf{x}).$$

As the solution might be ambiguous, we may rather be interested in determining any $\mathbf{x}^* \in \Sigma^d$ such that:

$$\sum_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)},\mathbf{x}^*) := \min_{\mathbf{x}\in\Sigma^d} \sum_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)},\mathbf{x}). \tag{2.40}$$

It turns out that such a vector can be determined easily. For that, we may use the following algorithm.

Algorithm 2.66. *To determine all solutions to Equation* (2.40)*, proceed as follows:*

1. *For $i = 1,\ldots,d$ do:*
   1.1. *Let $k_i := \max_{x\in\{x_i^{(1)},\ldots,x_i^{(n)}\}} \sum_{j\in[n]} \mathbf{1}(x = x_i^{(j)})$, i.e., the number of occurrences of the most frequently occurring character at index i;*
   1.2. *Let $E_i = \{x \in \{x_i^{(1)},\ldots,x_i^{(n)}\} : \sum_{j\in[n]} \mathbf{1}(x = x_i^{(j)}) = k_i\}$, i.e., the set of all characters that occur exactly k_i times at index i;*
2. *Return all $\mathbf{x}^* \in E_1 \times \cdots \times E_d$.*

Remark 2.67. If we are interested in any $\mathbf{x}^*$ which is a solution to Equation (2.40), then the above procedure may be implemented in such a way that, e.g., (a) it uses $O(|\Sigma| + d)$ additional memory units and $O(d|\Sigma| + dn)$ time (a bucket sort-like algorithm) or (b) with the usage of $O(n+d)$ additional memory units and $O(dn \log n)$ time, see also Remark 1.206.

Figure A.13 gives an exemplary C++ implementation which is based on hash tables and has an amortized run time of $O(d|\Sigma'| + dn)$, where $\Sigma' \subseteq \Sigma$ consists only of letters used in the input strings. Note that an input data set is given via a $d \times n$ integer matrix there.

Example 2.68. Let us set $d = 3$, $n = 6$, and $\Sigma = \{0,1,2,3\}$. Consider the following data set:

j	**1**	**2**	**3**	**4**	**5**	**6**
$s_1^{(j)}$	2	1	3	1	2	1
$s_2^{(j)}$	2	3	1	1	0	2
$s_3^{(j)}$	3	0	0	2	0	0

Noticing that $\min_{\mathbf{x}\in\Sigma^d} \sum_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)}, \mathbf{x}) = 9$, there are two solutions to Equation (2.40): $(1, 1, 0)$ and $(1, 2, 0)$. One of them is among the input vectors (this is not a rule in general), so it also corresponds to the set's medoid.

The median with respect to the Hamming distance is definitely symmetric, idempotent, internal, decomposable, and stable.

Remark 2.69. The above algorithm may easily be extended to find a weighted median, i.e., $\arg\min_{\mathbf{x}\in\Sigma^d} \sum_{i\in[n]} w_i \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)}, \mathbf{x})$, where $\mathbf{w}$ is a weighting vector.

2.7.2 Center

Now we shall focus on determining:

$$\mathsf{Center}_{\mathfrak{d}_{\mathrm{H}}}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \arg\min_{\mathbf{x}\in\Sigma^d} \bigvee_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)}, \mathbf{x}).$$

Again, the solution may be non-unique, therefore we rather shall be aiming at determining any $\mathbf{x}^* \in \Sigma^d$ such that:

$$\bigvee_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)}, \mathbf{x}^*) := \min_{\mathbf{x}\in\Sigma^d} \bigvee_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)}, \mathbf{x}). \tag{2.41}$$

Such a fusion function is of interest in coding theory [198], gene clustering [155], and other bioinformatics tasks [73].

Example 2.70. Let us go back to data in Example 2.68. There are two centers: $(1, 2, 0)$ and $(2, 1, 0)$. We have $\min_{\mathbf{x}\in\Sigma^d} \bigvee_{i\in[n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)}, \mathbf{x}) = 2$.

A center character sequence is at least idempotent, symmetric, and internal (or more precisely, there is at least one internal solution).

Unfortunately, there is no polynomial time-algorithm (with respect to n – it can be reduced to 3SAT) for computing it (unless P = NP) even for $d = 2$, see, e.g., [198, 305].

Among exact algorithms, which aim to find a string within some maximal distance threshold, e, that is $\mathbf{x}$ with $\bigvee_{i\in[n]} \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{x}) \leq e$, we may list [120, 231, 359], which are based on integer programming (IP), see [315]. One of the simplest formulations of the discussed problem may be written in terms of an IP task as follows (see [359]):

$$\text{minimize } \delta \quad \text{w.r.t. } \delta \in \mathbb{Z}, \mathbf{t} \in \mathbb{Z}^d, \mathbf{z} \in \mathbb{Z}^{n\times d} \tag{2.42}$$

subject to:

$$\delta - \sum_{j=1}^{d} z_{i,j} \quad \geq \quad 0, \quad i = 1, \ldots, n$$

$$\begin{aligned} t_j - kz_{i,j} &\leq s_j^{(i)}, \quad i = 1, \ldots, n, j = 1, \ldots, d \\ kz_{i,j} - t_j &\geq s_j^{(i)}, \quad i = 1, \ldots, n, j = 1, \ldots, d \\ \delta &\in \{0, \ldots, d\}, \\ t_j &\in \{1, \ldots, k\}, \quad j = 1, \ldots, d \\ z_{i,j} &\in \{0, 1\}, \quad i = 1, \ldots, n, j = 1, \ldots, d. \end{aligned}$$

Here we assume that $\Sigma = \{1, 2, \ldots, k\}$ (the original Σ may always be reencoded in such a way). The solution is stored in the $\mathbf{t}$ vector. This can be solved, e.g., using the COIN-OR SYMPHONY library (via the Rsymphony package in R). Please note that the above formulation leads to a practically unusable implementation (unless d, n are small).

Among other exact algorithms we may find the one given in [117]. Here, we start with a string in the input data set and then consecutively modify no more than e letters in the candidate string at a time. Another is given in [254]. It is based on some data reduction techniques and search tree algorithms. What is more, in [73] an algorithm to compute the closest string in the presence of outliers is given, i.e., one within a Hamming distance of δ to at least $n - k$ of the input strings for some k.

There are also polynomial-time approximation schemes, see, e.g., [305, 320, 357]. For instance, Lanctot et al. derive a polynomial-time $(4/3 + \varepsilon)$-approximation algorithm for any small $\varepsilon > 0$, see [305].

Remark 2.71. We say that a procedure is a $(1 + \varepsilon)$-approximation algorithm whenever the ratio of the *quality* of the result (here, expressed in terms of the Hamming distance) as compared to the optimal solution is guaranteed to be not greater than $1 + \varepsilon$ for any $\varepsilon > 0$.

As in practice exact algorithms exhibit poor performance, here let us discuss a so-called evolutionary strategy to approximate the center string. The first *genetic algorithms* were introduced by Fraser, see, e.g., [201, 202]. This is a class of adaptive, approximate optimization algorithms inspired by the biological process of natural selection. Of course, they do not guarantee that the global maximum of a fitness function f shall be found. However, such techniques are especially useful if the objective function is defined on a discrete space (and this is our case).

Algorithm 2.72. *For a given k (population size), η (number of iterations), and some fit measure f:*

1. *Generate a random initial population, i.e., a set of k initial elements (individuals) $P = \{\mathbf{p}^{(1)}, \ldots, \mathbf{p}^{(k)}\}$;*
2. *Determine the fit measure f_j for each individual $\mathbf{p}^{(j)}$;*
3. *Set P to be the best population considered so far, P^*;*
4. *For $i = 1, \ldots, \eta$ do:*

4.1. Selection: randomly select k pairs of vectors in P in, e.g., such a way that each vector occurs in the resulting sample with probability proportional to some function of f_j (fitness proportionate selection);

4.2. Crossover: generate a new population P', such that each new individual is obtained by combining elements in a pair of vectors selected in the previous step (some characters are taken from the first vector in a pair, the other ones are taken from the second vector);

4.3. Mutation: replace a few randomly chosen elements of vectors in P' with some other characters;

4.4. Set $P := P'$ and recompute the fit measures f_j, $j = 1, \ldots, k$;

4.5. If the current population includes an individual of the best fit so far, i.e., $\max_j f_j > \max_j f_j^*$, set $P^* := P$;

5. Return the best individual from P^*, i.e., $\arg\min_{\mathbf{p}^{*(j)}, j \in [k]} f_j^*$.

In our case, the fit measure of an individual $\mathbf{p}^{(j)}$ is inversely proportional to $\bigvee_{i \in [n]} \mathfrak{d}_{\mathrm{H}}(\mathbf{x}^{(i)}, \mathbf{p}^{(j)})$. Figure A.15 gives an exemplary R implementation of Algorithm 2.72, which aims to determine an approximate solution to Equation (2.41). Note that the crossover scheme choice is crucial here: we observe that the uniform crossover works far better than its one- or two-point version. There are some other possible options too, e.g., a crossover based on three parents or different selection phase schemes.

Table 2.4. ASCII codes (Unicode block *C0 Controls and Basic Latin*) and their corresponding code points (**char**s). **dec** stands for a decimal code and **bit** gives its corresponding 7-bit sequence.

dec	bit	char	dec	bit	char	dec	bit	char
0	0000000	*NUL*	43	0101011	+	86	1010110	V
1	0000001	*SOH*	44	0101100	,	87	1010111	W
2	0000010	*STX*	45	0101101	-	88	1011000	X
3	0000011	*ETX*	46	0101110	.	89	1011001	Y
4	0000100	*EOT*	47	0101111	/	90	1011010	Z
5	0000101	*ENQ*	48	0110000	0	91	1011011	[
6	0000110	*ACK*	49	0110001	1	92	1011100	\
7	0000111	*BEL*	50	0110010	2	93	1011101	]
8	0001000	*BS*	51	0110011	3	94	1011110	^
9	0001001	*HT*	52	0110100	4	95	1011111	_
10	0001010	*LF*	53	0110101	5	96	1100000	‘
11	0001011	*VT*	54	0110110	6	97	1100001	a
12	0001100	*FF*	55	0110111	7	98	1100010	b
13	0001101	*CR*	56	0111000	8	99	1100011	c
14	0001110	*SO*	57	0111001	9	100	1100100	d
15	0001111	*SI*	58	0111010	:	101	1100101	e
16	0010000	*DLE*	59	0111011	;	102	1100110	f
17	0010001	*DC1*	60	0111100	<	103	1100111	g
18	0010010	*DC2*	61	0111101	=	104	1101000	h
19	0010011	*DC3*	62	0111110	>	105	1101001	i
20	0010100	*DC4*	63	0111111	?	106	1101010	j
21	0010101	*NAK*	64	1000000	@	107	1101011	k
22	0010110	*SYN*	65	1000001	A	108	1101100	l
23	0010111	*ETB*	66	1000010	B	109	1101101	m
24	0011000	*CAN*	67	1000011	C	110	1101110	n
25	0011001	*EM*	68	1000100	D	111	1101111	o
26	0011010	*SUB*	69	1000101	E	112	1110000	p
27	0011011	*ESC*	70	1000110	F	113	1110001	q
28	0011100	*FS*	71	1000111	G	114	1110010	r
29	0011101	*GS*	72	1001000	H	115	1110011	s
30	0011110	*RS*	73	1001001	I	116	1110100	t
31	0011111	*US*	74	1001010	J	117	1110101	u
32	0100000	*(space)*	75	1001011	K	118	1110110	v
33	0100001	!	76	1001100	L	119	1110111	w
34	0100010	"	77	1001101	M	120	1111000	x
35	0100011	#	78	1001110	N	121	1111001	y
36	0100100	$	79	1001111	O	122	1111010	z
37	0100101	%	80	1010000	P	123	1111011	{
38	0100110	&	81	1010001	Q	124	1111100	|
39	0100111	’	82	1010010	R	125	1111101	}
40	0101000	(	83	1010011	S	126	1111110	~
41	0101001	)	84	1010100	T	127	1111111	*DEL*
42	0101010	*	85	1010101	U			

Chapter 3

Aggregation of strings

Up to now we have discussed different data fusion frameworks in the case of numeric (quantitative), ordinal, and nominal data. We started with a mathematically simple case of unidimensional data and then considered a more complex setting in which the aggregated objects were tuples of length d. In other words, for a given set X we considered:

— $\mathsf{F} : X^n \to X$ (univariate fusion functions, see Chapter 1),

— $\mathsf{F} : (X^d)^n \to X^d$ (d-variate fusion functions, see Chapter 2).

The above frameworks can be extended so that fusion functions which take an arbitrary number of elements as input are obtained:

— $\mathsf{F} : X^* \to X$ (extended univariate fusion functions),

— $\mathsf{F} : (X^d)^* \to X^d$ (extended d-variate fusion functions).

Surely, each extended fusion function may be conceived of as a family of fusion functions, each acting on tuples of different lengths.

Remark 3.1. Aggregation theoreticians sometimes also consider fusion functions which act on infinite sequences of elements. This is useful for studying asymptotic behavior of fusion functions, see, e.g., [223, 363]. As this kind of data does not occur in computational tasks (given a data set, one may always determine $d^+ = \max\{|\mathbf{x}^{(i)}|, i = 1, \ldots, n\}$), we do not discuss such a framework in this monograph.

It turns out that one more type of extension may be useful. Namely, we can be interested in aggregating vectors of arbitrary (nonconforming) lengths. Such a scenario from now on is called fusion of *strings*, *anyvariate* or *variable length data*. More specifically:

— $\mathsf{F} : (X^*)^n \to X^*$ (n-ary fusion functions to aggregate vectors of any length),

— $\mathsf{F} : (X^*)^* \to X^*$ (extended fusion functions to aggregate an arbitrary number of vectors of any length).

Depending on the choice of X, this situation frequently occurs, e.g., in the case of informetric data ($X = \mathbb{I}$) or character strings (like DNA or bit sequences – nominal scale).

Remark 3.2. A sequence of strings with elements being real numbers may be represented in R/Rcpp as a `List` object (a vector of elements of any type), which stores `NumericVectors` as its elements. In the case of character strings, see Section 3.2, this corresponds to the `CharacterVector` type, whose elements are objects of class `Rcpp::String`. In pure C we may use such data types as `double**` and `char**`, respectively, and, when using the C++ Standard Library (or STL) objects, we may set `std::vector< std::vector<double> >` and `std::vector< std::string >`, respectively.

Having in mind that the current space is even more "complex" than the previous one, this time let us begin with a review of different orderings on X^*.

3.1 Orders in the space of strings

Let $X^* = \bigcup_{d=1}^{\infty} X^d$ (note that this time we include vectors of length one). In a parallel section from the previous chapter we studied – among others – the so-called product order, which was a way to extend a partial or linear ordering relation $\sqsubseteq$ on some set P to the case of P^d for some d. Here we are naturally interested in a review of different ways of extending $\sqsubseteq$ to $\sqsubseteq^*$ in such a way that it may act on P^*. More formally, given a poset $\mathcal{P} = (P, \sqsubseteq)$, our aim is to construct $\mathcal{P}^* = (P^*, \sqsubseteq^*)$. It turns out that several interesting ways to do so exist in the aggregation literature.

3.1.1 Lexicographic order

The *lexicographic order* is defined for $\mathbf{p}, \mathbf{q} \in P^*$ as:

$$\mathbf{p} \sqsubseteq^* \mathbf{q} \iff (\exists i \in [d_p \wedge d_q])(\forall j \in [i-1])\ p_j = q_j \text{ and } p_i \sqsubset q_i \text{ if } i < d_p \text{ and } p_i \sqsubseteq q_i \text{ otherwise,} \tag{3.1}$$

where $|\mathbf{p}| = d_p$, $|\mathbf{q}| = d_q$ and, as usual, $p \sqsubset q$ whenever $p \sqsubseteq q$ with $p \neq q$.

Note that if $\sqsubseteq$ is a linear order, then also $\sqsubseteq^*$ is one. Moreover, if $\mathcal{P}$ is bounded from below with the least element denoted with $\underline{0}$, then $(\underline{0})$ is the least element of $\mathcal{P}^*$.

Example 3.3. Let $P = \{\mathtt{a}, \mathtt{b}, \ldots, \mathtt{z}\}$ and $\mathtt{a} \sqsubset \mathtt{b} \sqsubset \cdots \sqsubset \mathtt{z}$. In such a case, we have $\texttt{"a"} \sqsubset^* \texttt{"aa"} \sqsubset^* \texttt{"aaa"} \sqsubset^* \cdots \sqsubset^* \texttt{"ab"} \sqsubset^* \texttt{"aba"} \sqsubset^* \texttt{"abaa"} \sqsubset^* \cdots \sqsubset^* \texttt{"abb"} \sqsubset^* \cdots \sqsubset^* \texttt{"b"}$ etc., where, e.g., $\texttt{"abc"} = (\mathtt{a}, \mathtt{b}, \mathtt{c})$.

Remark 3.4. Lexicographic order determines exactly how character strings are ordered in many locales. Yet, in natural language processing tasks there are some exceptions to this rule. For instance, in the Slovak locale (similar rules exist for Czech), we have $\texttt{"citlivý"} \sqsubset^* \texttt{"hladný"} \sqsubset^* \texttt{"chladný"}$ (delicate / hungry / cool). This is because *ch* is treated as a digraph here and in fact it should be treated as a distinct, single character. Moreover, if we compare strings which consist of numerals, one might need a different order here, e.g., one that gives $\texttt{"ID\_69"} \sqsubset^* \texttt{"ID\_123"}$; please refer to the Unicode Technical Standard on string collation [136] for more information.

Among modified versions of the lexicographic ordering we find, among others, the Luzin-Sierpiński (Kleene-Brouwer) order, which gives a greater priority to a string with prefix $\mathbf{p}$ than to the sole string $\mathbf{p}$ in its entirety, see [275].

3.1.2 α- and β-, and informetric orderings

The so-called α- and β-orderings were introduced by Carbonell, Mas, and Mayor in [101] for the purpose of studying extended classical aggregation functions and constructing weighting triangles, compare Section 1.4.1. Moreover, they were considered in a more general (lattice) setting by Calvo and Mayor in [98]. Assuming that $(P, \sqsubseteq, \sqcap, \sqcup)$ is a complete lattice, we have what follows.

Definition 3.5. Let $\mathbf{p} \in P^{d_p}, \mathbf{q} \in P^{d_q}$. Then $\mathbf{p} \sqsubseteq_\alpha \mathbf{q}$ if and only if $d_p \leq d_q$ and $(\forall i \in [d_p])\ p_i \sqsubseteq q_i$ and if additionally $d_p < d_q$, then $\bigsqcup_{i=1}^{d_p} p_i \sqsubseteq \bigsqcap_{i=d_p+1}^{d_q} q_i$.

Definition 3.6. Let $\mathbf{p} \in P^{d_p}, \mathbf{q} \in P^{d_q}$. Then $\mathbf{p} \sqsubseteq_\beta \mathbf{q}$ if and only if $d_p \geq d_q$ and $(\forall i \in [d_q])\ p_i \sqsubseteq q_i$ and if additionally $d_p > d_q$, then $\bigsqcup_{i=d_q+1}^{d_p} p_i \sqsubseteq \bigsqcap_{i=1}^{d_q} q_i$.

We see that for $d_p = d_q$ both orders coincide with the extension of the product order to P^*, $\sqsubseteq^*$, defined as $\mathbf{p} \sqsubseteq^* \mathbf{q}$ whenever $d_p = d_q$ and $\mathbf{p} \sqsubseteq^d \mathbf{q}$. Formally, as each binary relation on X is in fact a subset of X^2, we have that $\sqsubseteq^* \subseteq \sqsubseteq_\alpha$ and $\sqsubseteq^* \subseteq \sqsubseteq_\beta$.

If $(P, \sqsubseteq, \sqcap, \sqcup, \underline{0}, \overline{1})$ is a bounded lattice, then $\underline{0}$ is the least element with respect to $\sqsubseteq_\alpha$ and $\overline{1}$ is the greatest one with respect to $\sqsubseteq_\beta$.

A somehow more relaxed version of $\sqsubseteq_\alpha$ may be formulated as follows.

Definition 3.7. Let $\mathbf{p} \in P^{d_p}, \mathbf{q} \in P^{d_q}$. Then $\mathbf{p} \sqsubseteq_\gamma \mathbf{q}$ if and only if $d_p \leq d_q$ and $(\forall i \in [d_p])\ p_i \sqsubseteq q_i$.

This type of ordering is useful in informetric tasks, see the next section for details. If $(P, \sqsubseteq, \sqcap, \sqcup, \underline{0}, \overline{1})$ is a bounded lattice, then $\underline{0}$ is the least element with respect to $\sqsubseteq_\gamma$.

Proposition 3.8. *Given a bounded lattice $(P, \sqsubseteq, \sqcap, \sqcup, \underline{0}, \overline{1})$, we have what follows for every $\mathbf{p} \in P^d$ and $d \in \mathbb{N}$:*

— $(p_1, \ldots, p_d, \bigsqcap_{i=1}^{d} p_i) \sqsubseteq_\beta (p_1, \ldots, p_d) \sqsubseteq_\alpha (p_1, \ldots, p_d, \bigsqcup_{i=1}^{d} p_i)$, *see [98],*

— $(p_1) \sqsubseteq_\gamma (p_1, p_2) \sqsubseteq_\gamma \cdots \sqsubseteq_\gamma (p_1, \ldots, p_d) \sqsubseteq_\gamma (p_1, \ldots, p_d, \underline{0})$.

Also, if $(P, \sqsubseteq)$ is a chain, then the above extensions of $\sqsubseteq$ generate lattices.

Remark 3.9. Regarding classical fusion functions based on the above orderings, we have what follows. Let $\mathsf{F} : P^* \to P$ be a fusion function monotonic with respect to $\sqsubseteq^*$. Then:

— F is monotonic with respect to $\sqsubseteq_\alpha$ if and only if $\mathsf{F}(\mathbf{p}) \sqsubseteq \mathsf{F}(\mathbf{p}, \bigsqcup_{i=1}^{d_p} p_i)$,

— F is monotonic with respect to $\sqsubseteq_\beta$ if and only if $\mathsf{F}(\mathbf{p}, \bigsqcap_{i=1}^{d_p} p_i) \sqsubseteq \mathsf{F}(\mathbf{p})$

for all $\mathbf{p} \in P^*$, see [98].

Notably, in [98] the concept of an extended aggregation function on P^* has been defined with the requirement of idempotency, as well as α-, and β-monotonicity.

Also let $\mathsf{F} : P^* \to P$ be a fusion function monotonic with respect to $\sqsubseteq^*$ and $(P, \sqsubseteq, \underline{0}, \overline{1})$ be a bounded poset. Then F is monotonic with respect to $\sqsubseteq_\gamma$ if and only if $\mathsf{F}(\mathbf{p}) \sqsubseteq \mathsf{F}(\mathbf{p}, \underline{0})$.

Yet, in this chapter we are interested in fusion functions like $\mathsf{F} : (P^*)^n \to P^*$.

3.1.3 Aggregation methods

As by using the listed extensions of $\sqsubseteq$ we get different lattices, trivially, methods already discussed (note their great generality) in Section 1.7 may be used to aggregate such kinds of data.

3.2 Aggregation of informetric data

Typical practical situations in which we are faced with the need to aggregate vectors of any length with elements in some real interval $\mathbb{I}$ (commonly $\mathbb{I} = [0, \infty]$

Table 3.1. Representative instances of informetric and similar data, where numeric lists of nonconforming lengths may be encountered, see, e.g., [107].

producer	**products**	**rating method**
R package author	R packages	Number of dependencies
Developer team	Python packages	Number of namespace imports from other projects
Web server	Web pages	Number of targeting weblinks or Page Rank
Web service server	JSON/XML-RPC methods	Number of remote procedure calls
Developer team	Code repository (git, svn, etc.)	Number of commits or lines of code
Publisher	On-line document	Number of downloads
Social networking profile	Posts	Number of "tweets" or "likes"
StackOverflow users	Answers to other users' questions	Up-votes
YouTube channels	Videos	Number of views
Digital library	Subscriber	Number of accesses
Scientist	Scientific articles	Number of citations
Scientific institute	Scientists	The h-index
Factory	Model-ranges of products	Sale results
Factory product	Supplied lots	Number of items without defects
Artist	Paintings	Auction price

or $\mathbb{I} = [-\infty, \infty]$) include scientometrics, webometrics, marketing, manufacturing, or quality engineering. Such application domains are sometimes referred to as *informetrics* (information metrics), and their aim is to deal with quantitative aspects of information processes. Here we assume that we have a set of abstract *producers* that output various numbers of *products* and each product is given a numeric valuation, representing its quality, see Figure 3.1, Table 3.1, and, e.g., [107, 199, 214, 215].

Most often, the order of elements in input vectors does not matter. Therefore, we may assume that the vectors we aggregate are already sorted. For any d and a fixed $\mathbb{I}$, let $\mathcal{S}_d$ designate the set of nonincreasingly ordered vectors of length d, i.e., $\mathcal{S}_d = \{(x_1, \ldots, x_d) \in \mathbb{I}^d, x_1 \geq \cdots \geq x_d\}$. Moreover, let $\mathcal{S}_{\leq d}$ be a set of nonincreasingly ordered vectors of length at most d, that is $\mathcal{S}_{\leq d} = \bigcup_{i=1}^{d} \mathcal{S}_i$.

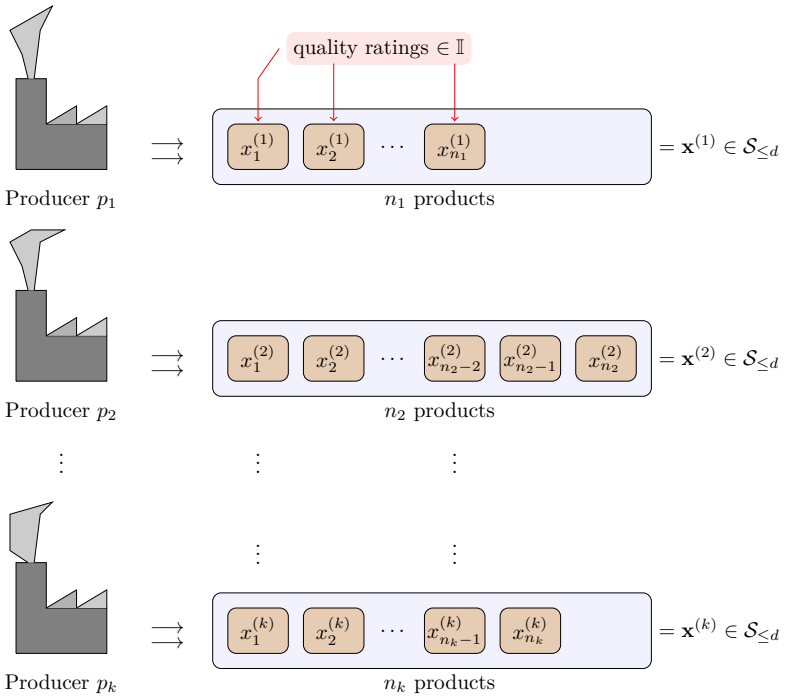


Figure 3.1. Producers, products, and their quality ratings, see [107].

From now on we also assume that $\mathcal{S} = \mathcal{S}_{\leq \infty}$.

Suppose that we are given n producers and that each of them produced no more than d products for some d. Obviously, such d is finite and well defined for each set of producers. The set of producers may thus be represented as $\mathcal{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$, where $\mathbf{x}^{(i)} = \left(x_1^{(i)}, \ldots, x_{d_i}^{(i)}\right) \in \mathcal{S}_{\leq d}$ for all $i = 1, \ldots, n$. For instance, $x_j^{(i)}$ may denote the number of citations of the jth most cited paper of the ith scholar, or the score of the jth best post written by the ith Stack Exchange user.

In this section we are interested in constructing fusion functions like $\mathsf{F} : \mathcal{S}_{\leq d}^n \to \mathcal{S}_{\leq d}$ or their extended versions $\mathsf{F} : \mathcal{S}_{\leq d}^* \to \mathcal{S}_{\leq d}$. They aim to determine the most "typical" or "representative" output of a producer in a cluster of producers. They may be used in, among others, informetric data clustering tasks, see the papers by Cena and Gagolewski [105, 106] (for a (fuzzy) k-means-like procedure) and also [118, 130, 256, 380]. Moreover, note that in Chapter 5 we shall focus on numeric characteristics of informetric data, which include such tools as the famous Hirsch h-index (a particular Sugeno integral).

Possibly desirable properties of fusion functions of our interest here include:

— monotonicity with respect to $\sqsubseteq_\gamma$,

— symmetry,

— $\mathsf{F}(n * \mathbf{x}) = \mathbf{x}$, (idempotency)

— $\mathsf{F}((x_1), (x_1, x_2), \ldots, (x_1, x_2, \ldots, x_n)) = (x_1, x_2, \ldots, x_j)$ for some $j \in [n]$ and all n as well as $x_1, x_2, \ldots x_n \in \mathbb{I}$, $x_i \geq x_{i+1}$, $i \in [n-1]$, (idempotency on common indices)

— $d_y \in [d_{\min}, d_{\max}]$, (length internality)

— $y_j \in [\bigwedge_{i=1}^{n} x_j^{(i)}, \bigvee_{i=1}^{n} x_j^{(i)}]$ for all $j \in [d_{\min}]$, (componentwise internality on common indices)

— $y_j \in [\bigwedge_{i=1}^{n} x_{d_i}^{(i)}, \bigvee_{i=1}^{n} x_1^{(i)}]$ for every $j \in [d_y]$, (global internality)

— $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathsf{F}(\mathbf{x'}^{(1)}, \ldots, \mathbf{x'}^{(n)}) = \mathbf{y}$ where $\mathbf{x'}^{(i)} \in \{\mathbf{x}^{(i)}, \mathbf{y}\}$ for every $i \in [n]$,

— stability,

for all $\mathbf{x}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in \mathcal{S}_{\leq d}$, where we assume that $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathbf{y}$, $d_y = |\mathbf{y}|$, $d_i = |\mathbf{x}^{(i)}|$, $d_{\min} = \bigwedge_{i=1}^{n} d_i$, and $d_{\max} = \bigvee_{i=1}^{n} d_i$.

3.2.1 Metrics on the space of numeric strings

In order to construct fusion functions on the considered domain, defined as minimizers of some penalty, let us study a family of metrics introduced by Cena, Gagolewski, and Mesiar in [108]. Recall that among some interesting metrics in the space of vectors of the same lengths $\mathbb{R}^d$, we have, e.g., the Euclidean $\mathfrak{d}_2(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$ or the Manhattan $\mathfrak{d}_1(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} |x_i - y_i|$ distance. Metrics on sets of real vectors are most often defined by considering absolute values of pairwise differences of vectors' elements (see the notion of a norm-generated metric). Letting $\sum_{i=u}^{v} \cdots = 0$ for $u > v$, one way to redefine, e.g., $\mathfrak{d}_1$ so that it acts on elements in $\mathcal{S}_{\leq d}$ is to consider:

$$\mathfrak{d}'_1(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d_x} |x_i - y_i| + \sum_{i=d_x+1}^{d_y} |y_i|, \tag{3.2}$$

where, by symmetry, without loss in generality, we assume that $d_x \leq d_y$. Note that as $|a - 0| = |0 - a| = |a|$, we have $\mathfrak{d}'_1(\mathbf{x}, \mathbf{y}) = \mathfrak{d}_1(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$, where, e.g., $\tilde{\mathbf{x}} = (x_1, \ldots, x_{d_x}, 0, 0, \ldots, 0) \in \mathbb{R}^d$, i.e., a version of $\mathbf{x}$ padded with 0s (a similar idea is reflected in the Hirsch h-index, which in fact is the Ky Fan metric [183] applied to $\mathbf{x}$ and a $\mathbf{0}$ vector of the same length as $\mathbf{x}$).

Unfortunately, $\mathfrak{d}'_1$ is only a pseudometric on $\mathcal{S}_{\leq d}$: a vector $(x_1, \ldots, x_{d_x})$ is indistinguishable from $(x_1, \ldots, x_{d_x}, 0, 0, \ldots, 0)$. In other words, nonexistent products are treated in the same way as products of quality 0. This setting, however, is not completely valid in our framework. Thus, an additional penalty for the difference in vectors' lengths may be introduced.

Theorem 3.10. *Let $\mathfrak{d} : \mathcal{S}_{\leq d} \times \mathcal{S}_{\leq d} \to [0, \infty]$ be such that $\mathfrak{d}(\mathbf{x}, \mathbf{y}) = \mu(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) + \nu(\mathbf{x}, \mathbf{y})$, where μ is a metric on $\mathbb{R}^d$ and ν is a pseudometric on $\mathcal{S}_{\leq d}$. Then $\mathfrak{d}$ is a metric on $\mathcal{S}_{\leq d}$ if and only if for all $\mathbf{x}, \mathbf{y}$ such that $\tilde{\mathbf{x}} = \tilde{\mathbf{y}}$ it holds $\nu(\mathbf{x}, \mathbf{y}) = 0 \implies n_x = n_y$.*

In particular, we may consider $\nu(\mathbf{x}, \mathbf{y})$ which is just a function of vector lengths, e.g., $\nu(\mathbf{x}, \mathbf{y}) = p|d_x^r - d_y^r|$ for any $p, r > 0$. In such a way, the Manhattan and the Euclidean metric may be rewritten as:

$$\mathfrak{d}_{M_1,p,r}(\mathbf{x}, \mathbf{y}) = \mathfrak{d}_1(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) + p|d_x^r - d_y^r| \tag{3.3}$$

and:

$$\mathfrak{d}_{M_2,p,r}(\mathbf{x}, \mathbf{y}) = \mathfrak{d}_2(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) + p|d_x^r - d_y^r|, \tag{3.4}$$

respectively. Take any vectors $\mathbf{x}, \mathbf{y}$ such that $n_x < n_y$. Both metric classes possess the important property that the distance between $\mathbf{x}$ and $\mathbf{y}$ is smaller than the distance between $\mathbf{x}$ and $(\mathbf{y}, a, a, \ldots, a)$, i.e., $\mathbf{y}$ padded with at least one value $a \in \mathbb{I}$. In other words, such metrics are able to distinguish vectors of different lengths from each other.

3.2.2 Centroid

Let us study fusion functions that minimize sums of $\mathfrak{d}_{M_2,p,r}$-based penalties of the form:

$$\begin{aligned} \mathfrak{d}_{p,r}^2(\mathbf{x}, \mathbf{y}) &= \sum_{i=1}^{\min\{d_x, d_y\}} (x_i - y_i)^2 + \sum_{i=d_x+1}^{n_y} y_i^2 + \\ &+ \sum_{i=d_y+1}^{n_x} x_i^2 + p|d_x^r - d_y^r|, \end{aligned} \tag{3.5}$$

which lead to centroid-like (see Section 2.5.3) mappings:

$$\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \arg\min_{\mathbf{y} \in \mathcal{S}} \sum_{i=1}^{n} \mathfrak{d}_{p,r}^2(\mathbf{x}^{(i)}, \mathbf{y}). \tag{3.6}$$

First of all, let us note that $|\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})| \leq d = \max\{|\mathbf{x}^{(i)}| : i = 1, \ldots, n\}$. It can be shown, see [106], that the value of F may be determined by using the following 2-step procedure.

1. First of all, for all $j = 1, \ldots, d$ we compute:

$$\mathbf{y}^{(j)} = \arg\min_{\mathbf{y} \in \mathcal{S}_d} \sum_{i=1}^{n} \mathfrak{d}_{p,r}^2(\mathbf{x}^{(i)}, \mathbf{y}).$$

2. Then we set:

$$\begin{aligned} j^* &= \arg\min_{j=1,\ldots,d} \sum_{i=1}^{n} \mathfrak{d}_{p,r}^2(\mathbf{x}^{(i)}, \mathbf{y}^{(j)}), \\ \mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) &= \mathbf{y}^{(j^*)}. \end{aligned}$$

It turns out that if we act on $\mathbb{I} = [0, \infty]$, we simply have for all $j \in [d]$ and $i \in [j]$:

$$y_i^{(j)} = \frac{1}{n} \sum_{k=1}^{n} \tilde{x}_i^{(k)}.$$

The obtained fusion function is symmetric, idempotent, and length internal, among others.

The above formula is unfortunately invalid for arbitrary $\mathbb{I}$. This is due to the fact that the space $\mathcal{S}_{\le d}$ consists of *ordered* vectors. Thus, in general, the task that aims to determine a penalty minimizer here is much more difficult. It may be shown that a procedure given in Figure A.17 may be used to compute $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$, see [106] for more details – some of the components have to be averaged. Unfortunately, if we allow negative elements, the resulting fusion function is definitely not $\sqsubseteq_\gamma$-nondecreasing.

Note that this procedure may relatively easily be generalized to the case of weighted $\mathfrak{d}_{p,r}^2$ penalty functions, see [105].

Example 3.11 ([106]). Let:

$$\mathcal{X} = \left\{ \begin{array}{lllllll} (& 42, & 21, & 12, & 10, & 8 &), \\ (& 1, & 0, & -10 & & &), \\ (& 0, & -1 & & & &), \\ (& -10, & -13 & & & &) \end{array} \right\}.$$

Assuming that $p = 1, r = 1$, we have $\mathsf{F}(\mathcal{X}) = (8\frac{1}{4}, 4\frac{1}{4}, 1\frac{2}{3}, 1\frac{2}{3}, 1\frac{2}{3})$. Here is the output of the procedure given in Figure A.17 for each $j = 1, \ldots, d$.

j	$\sum_{i=1}^{n} \mathfrak{d}_{1,1}^2(\mathbf{x}^{(i)}, \mathbf{y}^{(j)})$	$y_1^{(j)}$	$y_2^{(j)}$	$y_3^{(j)}$	$y_4^{(j)}$	$y_5^{(j)}$	$y_6^{(j)}$
1	3139.75	8.25					
2	3063.50	8.25	4.25				
3	3062.50	8.25	4.25	0.50			
4	3047.50	8.25	4.25	1.50	1.50		
5	<u>3034.17</u>	8.25	4.25	1.67	1.67	1.67	
6	3037.17	8.25	4.25	1.67	1.67	1.67	0.00

Example 3.12 ([106]). Let:

$$\mathcal{X} = \left\{ \begin{array}{lllllll} (& -10, & -12, & -14, & -16, & -17 &), \\ (& 1, & 0, & -10 & & &), \\ (& -10, & -15, & -16 & & &), \\ (& -20 & & & & &) \end{array} \right\}.$$

Then $\mathsf{F}(\mathcal{X}) = (-6.95, -6.95, -6.95, -6.95, -6.95)$ for $p = 1, r = 1$.

j	$\sum_{i=1}^{n} \mathfrak{d}_{1,1}^2(\mathbf{x}^{(i)}, \mathbf{y}^{(j)})$	$y_1^{(j)}$	$y_2^{(j)}$	$y_3^{(j)}$	$y_4^{(j)}$	$y_5^{(j)}$
1	1694.75	-9.750				
2	1528.50	-8.250	-8.250			
3	1126.50	-8.250	-8.250	-10.000		
4	1142.75	-7.625	-7.625	-7.625	-7.625	
5	<u>1108.95</u>	-6.950	-6.950	-6.950	-6.950	-6.950

3.2.3 1-Median

Due to the nature of the introduced metrics, the described 2-step procedure may also be incorporated in the case of finding the 1-median with respect to the $\mathfrak{d}_{M_1,p,r}$ and $\mathfrak{d}_{M_2,p,r}$ metrics.

For $\mathbb{I} = [0, \infty]$ and $\mathfrak{d}_{M_1,p,r}$, the 1-Median of course corresponds to the componentwise median (with missing elements treated as 0s). That is, for some $j \in [d]$ and $i \in [j]$ we have:

$$y_i^{(j)} = \mathsf{Median}(\tilde{x}_i^{(1)}, \ldots, \tilde{x}_i^{(n)}).$$

By the monotonicity of Median and the fact that $0 \leq x_i^{(j)}$ for all $j \in [n]$ and $i \in [d_j]$, we have that if $\mathbf{x}^{(j)} \in \mathcal{S}$, then $\mathbf{y}^{(j)} \in \mathcal{S}$. In other words, the resulting vector is surely sorted.

Remark 3.13. Inspired by the above derivations, we may introduce the following family of componentwise fusion functions for numeric strings in the case of $\mathbb{I} = [0, \infty]$. Let $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$ be a nondecreasing fusion function. Note that $\tilde{x}^{(1)}, \ldots, \tilde{x}^{(n)} \in \mathbb{I}^d$ and thus with data transformed in such a way we obtain a case exactly as in the previous chapter. Now let $\mathbf{y} \in \mathcal{S}_d$ be such that for $i \in [d]$ we have:

$$y_i = \mathsf{F}(\tilde{x}_i^{(1)}, \ldots, \tilde{x}_i^{(n)}).$$

Then a penalty-based solution may be given as $(y_1, \ldots, y_{d'})$ where d' is given by:

$$d' = \operatorname*{arg\,min}_{d' \in [d]} P\left((y_1, \ldots, y_{d'}); \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right).$$

For instance, P may be given as $P\left((y_1, \ldots, y_{d'}); \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right) = \sum_{k=1}^{n} p|d_k^r - d'^r|$ for some $p, r > 0$. Of course, similarly as in Definition 1.105, we require $P : \mathcal{S} \times \mathcal{S}^n \to [0, \infty]$ to fulfill:

— $P(\mathbf{y}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = 0$ if $\mathbf{y} = \mathbf{x}^{(j)}$ for all $j \in [n]$,

— for every fixed $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, the set of minimizers of $P(\mathbf{y}; \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$ is a singleton.

Moreover, if $\arg\min_{d' \in [d]} P\left((y_1, \ldots, y_{d'}); \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right)$ is ambiguous, then we may choose the smallest one or the largest d' which minimizes the penalty.

Note that the case of $\mathfrak{d}_{M_2,p,r}$ is slightly more difficult. First of all, we need the following result.

Proposition 3.14. *For all d and n, a convex combination of any n vectors in $\mathcal{S}_d$ is also a vector in $\mathcal{S}_d$.*

Of course, for any $\mathbf{x} \in \mathcal{S}_{\leq d}$ and $\mathbb{I} = [0, \infty]$, it holds that $\tilde{\mathbf{x}} \in \mathcal{S}_d$. Recall that in Section 2.5.1 we noted that the 1-median is within the convex hull of a set of input points, see Equation (2.27). This implies that $\mathbf{y}^{(j)}$ is sorted. Thus, e.g., the Weiszfeld algorithm may be used in the 1st step of our 2-step procedure.

3.3 Aggregation of character strings

This time, for a given X, let $X^* = \bigcup_{d=0}^{\infty} X^d$ denote the Kleene closure of X. In particular, an empty vector $\varepsilon \in X^*$.

In Section 3.2 we discussed a few methods to aggregate vectors of nonconforming lengths. Each member of such a vector was a real number. It was quite a comfortable situation, as algebraic operations like addition, multiplication, division, and so forth, were defined there – we were on an interval scale. Here we revisit a situation where vectors with elements on a nominal scale are to be aggregated, see Sections 1.8 and 2.7.

A tuple $\mathbf{x} \in \Sigma^*$ is often called a *character string* (over a finite set Σ), and an element s_i – the ith character. If, say, $\mathtt{a}, \mathtt{b} \in \Sigma$, then we shall sometimes write, e.g., `"aba"` instead of $(\mathtt{a}, \mathtt{b}, \mathtt{a})$.

Example 3.15. Going back to Example 1.202, character strings over $\Sigma = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$ may be interpreted as DNA sequences or protein sequences in the case of $|\Sigma| = 20$. On the other hand, referring to Example 1.203, we may also consider ASCII or Unicode character strings.

Example 3.16. Strings over $\Sigma = \{\mathtt{0}, \mathtt{1}\}$ are called bit strings. In fact, even if generally it is not the most convenient perspective, each computer file or a digital signal transmission may be viewed as sequence of bits (see Table 2.4). Interestingly, there are a few different ways to map Unicode code point sequences to bit sequences: UTF-8, UTF-16LE, UTF-16BE, UTF-32LE, UTF-32BE, and so on.

Here we are of course interested in fusion functions like $\mathsf{F} : (\Sigma^*)^n \to \Sigma^*$.

Remark 3.17. In SQL-oriented relational database management systems, the term "string aggregation" is usually understood as a form of string concatenation (joining). For instance, in SQLite, there is an aggregate `GROUP_CONCAT()`, which joins a given list of strings, separating each item with a comma. We have, e.g.:

$$\texttt{GROUP\_CONCAT}(\texttt{"a"}, \texttt{"bcd"}, \texttt{"abc"}) = \texttt{"a,bcd,abc"}.$$

A similar function in Oracle Database is named `LISTAGG()` and in PostgreSQL – `STRING_AGG()`. Additionally, it is worth noting that it is an associative string fusion function, see [313].

As far as desired properties of such fusion functions are concerned, firstly, let us stress that this time no ordering relation can naturally be taken into account, so we cannot rely on any type of monotonicity. On the other hand, at this point we may be expecting:

— symmetry,

— $\mathsf{F}(n * \mathbf{x}) = \mathbf{x}$, (idempotency)

— $|\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})| \in \left[\bigwedge_{i=1}^{n} |\mathbf{x}^{(i)}|, \bigvee_{i=1}^{n} |\mathbf{x}^{(i)}|\right]$, (length internality)

— $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) \in \Sigma'^{*}$ with $\Sigma' = \{s_j^{(i)} : i \in [n], j \in [|\mathbf{x}^{(i)}|]\}$ (the output is only based on characters which are used in the inputs; this is because we rather do not want the result of aggregating (`"a"`, `"c"`, `"c"`, `"a"`) to be `"b"`),

— for some $\mathtt{b}_1, \ldots, \mathtt{b}_j \in \Sigma$, there exists $i \in [j]$ such that $\mathsf{F}(\texttt{"}\mathtt{b}_1\texttt{"}, \texttt{"}\mathtt{b}_1\mathtt{b}_2\texttt{"}, \ldots, \texttt{"}\mathtt{b}_1\mathtt{b}_2 \ldots \mathtt{b}_j\texttt{"}) = \texttt{"}\mathtt{b}_1\mathtt{b}_2 \ldots \mathtt{b}_i\texttt{"}$,

— $\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \mathsf{F}(\mathbf{x}'^{(1)}, \ldots, \mathbf{x}'^{(n)}) = \mathbf{y}$ where $\mathbf{x}'^{(j)} \in \{\mathbf{x}^{(j)}, \mathbf{y}\}$,

— stability,

for each $\mathbf{x}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in \Sigma^{*}$.

The fusion functions discussed further on are defined as minimizers of carefully aggregated pseudometrics or metrics. Thus, in the latter case we naturally expect to obtain idempotent fusion functions. Nevertheless, we should not go that far now: firstly we shall define a few types of dissimilarity measures for strings.

3.3.1 Dissimilarity measures of character strings

Let us review the most frequently used dissimilarity measures of character strings, see also [77, 301, 371, 454]. They are not only used for constructing fusion functions, but have numerous other applications, e.g., in spelling correction [356, 368], error-tolerant pattern searching, fuzzy/approximate pattern matching [461], plagiarism detection [25, 26], text retrieval, optical character recognition, text clustering [265, 489], file revisions comparison (see the UNIX `diff` utility or software revision control systems like `git`, `svn`, or `mercurial`), and many others.

Below we discuss the following classes of character string distances:

— edit-based distances,

— q-gram-based distances,

— other.

Remark 3.18. Let us once again stress that some of the dissimilarity measures presented below – like the Jaccard q-gram index – are not necessarily "full" metrics. A few of them violate the condition "$\mathfrak{d}(x, y) = 0$ if and only if $x = y$". They are at least *pseudometrics*: the "only if" part of the axiom might not be met in all the cases.

A. Edit-based distances

Edit-based distances express the smallest total *cost* of necessary *changes* that have to be made to transform one string so as to get another one.

Definition 3.19. An *edit operation* is a pair $(\mathbf{a}, \mathbf{b}) \in \Sigma^* \times \Sigma^*$, written $\mathbf{a} \to \mathbf{b}$ for short.

For instance, the set $\mathbb{B} \subseteq \Sigma^* \times \Sigma^*$ of admissible edit operations may include:

— single character removal $((\forall \mathtt{a} \in \Sigma), (\texttt{"a"} \to \varepsilon) \in \mathbb{B})$,

— single character insertion $((\forall \mathtt{b} \in \Sigma), (\varepsilon \to \texttt{"b"}) \in \mathbb{B})$,

— single character substitution $((\forall \mathtt{a}, \mathtt{b} \in \Sigma), (\texttt{"a"} \to \texttt{"b"}) \in \mathbb{B})$,

— transposition of an adjacent pair of characters (swap; $(\forall \mathtt{a}, \mathtt{b} \in \Sigma)$, $(\texttt{"ab"} \to \texttt{"ba"}) \in \mathbb{B}$),

compare the historical Damerau paper [134] for discussion.

We may apply an edit operation $\mathbf{a} \to \mathbf{b}$ at a position i of a string $\mathbf{u}$, if:

$$(u_i, u_{i+1}, \ldots, u_{i+|\mathbf{a}|-1}) = \mathbf{a}.$$

In such a way, we derive a new string:

$$\mathbf{v} = (u_1, \ldots, u_{i-1}, \mathbf{b}, u_{i+|\mathbf{a}|}, \ldots, u_{|\mathbf{u}|}).$$

Given any two strings $\mathbf{u}, \mathbf{v}$, we may seek a *transforming sequence* $(s_1, \ldots, s_k)$ of edit operations and positions where they are applied, $s_i \in \mathbb{B} \times \mathbb{N}$, such that $\mathbf{v}$ may be derived step by step from $\mathbf{u}$. In order to define an edit distance, we shall assume that $\mathbb{B}$ is such that for each pair of strings there always exists at least one such sequence. From now on denote by $\mathrm{S}(\mathbf{u}, \mathbf{v}) \in (\mathbb{B} \times \mathbb{N})^*$ the set of all transforming sequences that enable us to derive $\mathbf{v}$ from $\mathbf{u}$.

Example 3.20. Assume that the set of admissible transformations consists only of single character removal and insertion. In such a setting, at least three steps are needed to derive `"fiction"` from `"function"`.

```
f u n c t i o n
  ↕ ..... "u" → ε (index 2)
f n c t i o n
  ↕ ..... "n" → ε (index 2)
f c t i o n
  ↕ ..... ε → "i" (index 2)
f i c t i o n
```

Also, we might be interested in introducing a function $\delta : \mathbb{B} \to \mathbb{R}_{0+}$ that measures the cost of applying any given edit operation.

Definition 3.21. A *generic edit distance* relative to a set of edit operations $\mathbb{B}$ and their costs $\delta : \mathbb{B} \to \mathbb{R}_{0+}$ is given for any $\mathbf{u}, \mathbf{v} \in \Sigma^*$ by:

$$\mathfrak{d}(\mathbf{u}, \mathbf{v}) = \min_{S \in \mathrm{S}(\mathbf{u},\mathbf{v})} \sum_{(b,i) \in S} \delta(b). \tag{3.7}$$

The following result follows from [77, Theorem 2.8].

Theorem 3.22. *Whenever $\mathbb{B}$ and δ are such that:*

- *if $(\mathbf{a} \to \mathbf{b}) \in \mathbb{B}$, then also the reverse operation $(\mathbf{b} \to \mathbf{a}) \in \mathbb{B}$; moreover, we have $\delta(\mathbf{a} \to \mathbf{b}) = \delta(\mathbf{b} \to \mathbf{a})$,*
- *if $(\mathbf{a} \to \mathbf{b}) \in \mathbb{B}$, then $\delta(\mathbf{a} \to \mathbf{b}) = 0$ implies that $\mathbf{a} = \mathbf{b}$,*
- *$\mathbb{B}$ is finite,*

then the generic edit distance relative to $\mathbb{B}$ and δ is a metric on $\Sigma^* \times \Sigma^*$.

Classical edit distances assume that each edit operation has unit cost.

Definition 3.23. The *longest common subsequence (LCS) distance* [372], $\mathfrak{d}_{\mathrm{LCS}}$, for $\mathbf{u}, \mathbf{v} \in \Sigma^*$ is defined as the minimal number of single character insertions and deletions that are used to derive $\mathbf{v}$ from $\mathbf{u}$. In other words, it is an edit distance given by $\mathbb{B} = \{(\texttt{"a"} \to \varepsilon), (\varepsilon \to \texttt{"a"}) : \mathtt{a} \in \Sigma\}$ and $\delta(b) = 1$ for every $b \in \mathbb{B}$.

If we additionally enable the use of a single character replacement operation, we might get an extended version of the metric introduced in Definition 2.65.

Remark 3.24. The extended *Hamming distance* $\mathfrak{d}_{\mathrm{H}}$ of $\mathbf{u}, \mathbf{v} \in \Sigma^*$ is given by:

$$\mathfrak{d}_{\mathrm{H}}(\mathbf{u}, \mathbf{v}) = \begin{cases} \sum_{i=1}^{d} \mathbf{1}(u_i \neq v_i) & \text{if } |\mathbf{u}| = |\mathbf{v}| = d, \\ \infty & \text{otherwise.} \end{cases} \tag{3.8}$$

It may may be conceived as a kind of edit distance, where single character replacement has unit cost and character insertion and removal has an infinitely large cost.

The LCS character modification scheme together with the replacement operation leads us also to the famous Levenshtein distance.

Definition 3.25. The *Levenshtein distance* [317], $\mathfrak{d}_{\mathrm{LV}}$, for $\mathbf{u}, \mathbf{v} \in \Sigma^*$ is defined as the minimal number of single character insertions, deletions, and replacements that are used to obtain $\mathbf{u}$ from $\mathbf{v}$, i.e., it is an edit distance defined by $\mathbb{B} = \{(\texttt{"a"} \to \varepsilon), (\varepsilon \to \texttt{"a"}), (\texttt{"a"} \to \texttt{"b"}) : \mathtt{a}, \mathtt{b} \in \Sigma\}$ and $\delta(b) = 1$ for every $b \in \mathbb{B}$.

Example 3.26. We have $\mathfrak{d}_{\mathrm{LV}}(\texttt{"function"}, \texttt{"fiction"}) = 2$.

```
f u n c t i o n
  ↕ ..... "u" → ε (index 2)
f n c t i o n
  ↕ ..... "n" → "i" (index 2)
f i c t i o n
```

It turns out that a dynamic programming scheme may be applied to calculate $\mathfrak{d}_{\mathrm{LV}}(\mathbf{u}, \mathbf{v})$, see, e.g., [458], as well as $\mathfrak{d}_{\mathrm{LCS}}(\mathbf{u}, \mathbf{v})$ [372], see also [457]. For instance, we have that $\mathfrak{d}_{\mathrm{LV}}(\mathbf{u}, \mathbf{v}) = d_{|\mathbf{u}|,|\mathbf{v}|}$, where $d_{0,0} = 0$, $d_{i,j} = \infty$ if $i \wedge j < 0$ and otherwise:

$$d_{i,j} = \min \left\{ \begin{array}{l} d_{i-1,j-1} + 1 \cdot \mathbf{1}(u_i \neq v_j), \\ d_{i,j-1} + 1, \\ d_{i-1,j} + 1. \end{array} \right\} \tag{3.9}$$

A basic algorithm runs in $O(|\mathbf{u}|\,|\mathbf{v}|)$ time and requires $O(|\mathbf{u}|\,|\mathbf{v}|)$ memory, which makes it practically unusable for long data streams (say, consisting of more than 100,000 characters). However, its advantage is that we may trace back the changes made in the first string to get the second string. If just the value of the edit distance is needed, only two rows of the $(d_{i,j})$ matrix need be allocated. This leads to the space complexity of $O(|\mathbf{u}| \wedge |\mathbf{v}|)$, see Figure A.18 for an exemplary implementation in the case of the Levenshtein distance. Please note that the proposed implementation acts on integer vectors, so for character strings it may be computed over Unicode text (all code points may be converted to UTF-32).

Another algorithm, given by Ukkonen [452], is able to compute $\mathfrak{d}_{\mathrm{LV}}(\mathbf{u}, \mathbf{v})$ in $O(d \cdot (|\mathbf{u}| \wedge |\mathbf{v}|))$ time and $O(d)$ space, where $d = \mathfrak{d}_{\mathrm{LV}}(\mathbf{u}, \mathbf{v})$. It may be modified to check in $O(t \cdot (|\mathbf{u}| \wedge |\mathbf{v}|))$ time whether $d \leq t$ for some t. A different algorithm, this time by Masek and Pateson, can be found in [352]: it leads to $O(n \max\{1, m/\log n\})$ time, where $n = (|\mathbf{u}| \vee |\mathbf{v}|)$ and $m = (|\mathbf{u}| \wedge |\mathbf{v}|)$.

If we additionally allow swapping any two adjacent characters, we get the following dissimilarity measure, formalized for the first time by Lowrance and Wagner in [329].

Definition 3.27. The *(unrestricted) Damerau–Levenshtein distance* $\mathfrak{d}_{\mathrm{DL}}$, for $\mathbf{u}, \mathbf{v} \in \Sigma^*$ is defined as the minimal number of single character insertions, deletions, replacements, and pairwise transpositions that are used to obtain $\mathbf{u}$ from $\mathbf{v}$, i.e., it is an edit distance given by $\mathbb{B} = \{(\texttt{"a"} \to \varepsilon), (\varepsilon \to \texttt{"a"}), (\texttt{"a"} \to \texttt{"b"}), (\texttt{"ab"} \to \texttt{"ba"}) : \mathtt{a}, \mathtt{b} \in \Sigma\}$ and $\delta(b) = 1$ for every $b \in \mathbb{B}$.

Also in this case there exists a dynamic programming approach-based algorithm, see [329], yet it is more complicated. We have $\mathfrak{d}_{\mathrm{DL}}(\mathbf{u}, \mathbf{v}) = d_{|\mathbf{u}|,|\mathbf{v}|}$, where $d_{0,0} = 0$, $d_{i,j} = \infty$ if $i \wedge j < 0$ and otherwise:

$$d_{i,j} = \min \left\{ \begin{array}{ll} d_{i-1,j-1} + 1 \cdot \mathbf{1}(u_i \neq v_j), & \\ d_{i,j-1} + 1, & \\ d_{i-1,j} + 1, & \\ \bigwedge\limits_{\substack{i'<i, j'<j \\ u_i = v_{j'} \text{ and } u_{i'} = v_j}} & (d_{i'-1,j'-1} + i - i' + j - j' - 1)\,. \end{array} \right\} \tag{3.10}$$

Remark 3.28. As noted by, e.g., Boytsov [77] and Loo [454], the (unrestricted) Damerau-Levenshtein distance is very often confused with its restricted version, namely the *optimal string alignment distance* (OSA). Informally, in OSA each substring is allowed to be edited only once. This dissimilarity measure is calculated as in Equation (3.10), but the minimum ($\wedge$) loop is computed only in the case of $i' = i - 1$ and $j' = j - 1$. For the unrestricted version, we for example have $\mathfrak{d}_{\mathrm{DL}}(\texttt{"ba"}, \texttt{"acb"}) = 2$.

```
b  a
↕ ..... "ba" → "ab" (index 1)
a  b
  ↕ ..... ε → "c" (edits an already modified part)
a  c  b
```

On the other hand, $\mathfrak{d}_{\mathrm{OSA}}(\texttt{"ba"}, \texttt{"acb"}) = 3$.

```
b  a
↕ ..... "b" → ε (index 1)
a
  ↕ ..... ε → "c" (index 2)
a  c
     ↕ ..... ε → "b" (index 3)
a  c  b
```

Also note that $\mathfrak{d}_{\mathrm{DL}}(\texttt{"ba"}, \texttt{"ab"}) = \mathfrak{d}_{\mathrm{OSA}}(\texttt{"ba"}, \texttt{"ab"}) = 1$ and $\mathfrak{d}_{\mathrm{DL}}(\texttt{"ab"}, \texttt{"acb"}) = \mathfrak{d}_{\mathrm{OSA}}(\texttt{"ab"}, \texttt{"acb"}) = 1$. We see that OSA does not fulfill the triangle inequality and hence it is not a metric on $\Sigma^* \times \Sigma^*$.

As we already mentioned, classical edit distances assume that each edit operation has unit cost. However, some software libraries like the stringdist [454]

package for R allow us to specify costs of each type of edit operation. For instance, having been given $w_{\mathrm{I}}, w_{\mathrm{D}}, w_{\mathrm{R}} > 0$, i.e., costs of insertion, deletion, and replacement, respectively, we may set $(\forall \mathtt{a}, \mathtt{b} \in \Sigma)$ $\delta(\mathtt{"a"} \to \varepsilon) = w_{\mathrm{D}}$, $\delta(\varepsilon \to \mathtt{"B"}) = w_{\mathrm{I}}$, $\delta(\mathtt{"a"} \to \mathtt{"b"}) = w_{\mathrm{R}}$. In such a way we get a weighted Levenshtein distance, see [294]. It is easily seen that this leads to the following modification of Equation (3.9):

$$d_{i,j} = \min \left\{ \begin{array}{l} d_{i-1,j-1} + w_{\mathrm{R}} \cdot \mathbf{1}(u_i \neq v_j), \\ d_{i,j-1} + w_{\mathrm{I}}, \\ d_{i-1,j} + w_{\mathrm{D}}. \end{array} \right\}$$

According to Theorem 3.22, a weighted Levenshtein distance is a metric if $w_{\mathrm{I}} = w_{\mathrm{D}}$. Moreover, if $w_{\mathrm{R}} = w_{\mathrm{I}} + w_{\mathrm{D}}$, then a dissimilarity measure proportional to the longest common subsequence distance is obtained.

Remark 3.29. More complex edit operations and cost dispatch schemes are suitable for natural language processing tasks (e.g., automated spell checking). For instance, in the case of the German language, we may set $\delta(\mathtt{"ß"} \to \mathtt{"ss"})$ to be smaller than the cost of other replacement operations etc.

It is worth noting that various modifications of edit distances exist in the literature. In particular, a constrained version of the Levenshtein distance, with limits on the number and type of edit operations performed, was proposed by Oommen in [379]. Moreover, Marzal and Vidal, see [351], discuss a normalized edit distance defined as the minimum with respect to all transforming sequences $\mathrm{S}(\mathbf{u}, \mathbf{v})$ of w/p, where w is the sum of unit costs of edit operations in a transforming sequence and p stands for the number of such operations. Additionally, in [404] an algorithm for edit-distance learning (more precisely, determining costs of edit operations) is given.

B. Q-gram-based distances

Given a string $\mathbf{u}$, a *q-gram*, $|\mathbf{u}| \geq q \geq 1$, is its substring that consists of q consecutive characters in $\mathbf{u}$, see [453]. The concept dates back to Shannon [422] and was used by him to model processes generating discrete sequences of characters. Nowadays, q-grams (at a word level[1]) are used, among others for automated search engine query completion.

Let $\mathcal{Q}_q(\mathbf{u})$ denote the set of all q-grams in $\mathbf{u}$, that is:

$$\mathcal{Q}_q(\mathbf{u}) = \Big\{ (u_i, u_{i+1}, \ldots, u_{i+q-1}) \in \Sigma^q : i = 1, \ldots, |\mathbf{u}| - q + 1 \Big\}. \tag{3.11}$$

We see that $|\mathcal{Q}_q(\mathbf{u})| = |\mathbf{u}| - q + 1$.

[1]For instance, Google in 2006 released a 24 GB compressed data set which consists of 13,588,391 unique words; we may find, e.g., 1,176,470,663 5-grams, ranked by the frequency of their occurrences, see `https://catalog.ldc.upenn.edu/LDC2006T13`.

Example 3.30. Three bigrams can be obtained from `"ACTG"`: it holds $\mathcal{Q}_2(\texttt{"ACTG"}) = \{\texttt{"AC"}, \texttt{"CT"}, \texttt{"TG"}\}$.

It turns out that q-grams may be used to define dissimilarity measures for strings.

Definition 3.31. Given $\mathbf{u}, \mathbf{v} \in \Sigma^*$ and $q \leq |\mathbf{u}| \wedge |\mathbf{v}|$, the *Jaccard q-gram dissimilarity index* is given by:

$$\mathfrak{d}_{\mathrm{J},q}(\mathbf{u}, \mathbf{v}) = 1 - \frac{|\mathcal{Q}_q(\mathbf{u}) \cap \mathcal{Q}_q(\mathbf{v})|}{|\mathcal{Q}_q(\mathbf{u}) \cup \mathcal{Q}_q(\mathbf{v})|} \in [0, 1]. \tag{3.12}$$

Remark 3.32. It holds $\mathfrak{d}_{\mathrm{J},2}(\texttt{"aab"}, \texttt{"aaab"}) = 0$, thus a Jaccard index is not a metric. The property $\mathfrak{d}_{\mathrm{J},q}(\mathbf{u}, \mathbf{v}) = 0 \iff \mathbf{u} = \mathbf{v}$ is violated for all pairs of strings with non-unique q-grams. However, a Jaccard index is positive, symmetric, and fulfills the triangle inequality and hence it is a pseudometric.

Let $c_{\mathbf{q}}(\mathbf{u})$ designate the number of occurrences of a substring $\mathbf{q}$ in $\mathbf{u}$:

$$c_{\mathbf{q}}(\mathbf{u}) = \left| \left\{ i = 1, \ldots, |\mathbf{u}| - |\mathbf{q}| + 1 : (u_i, \ldots, u_{i+|\mathbf{q}|-1}) = \mathbf{q} \right\} \right|. \tag{3.13}$$

Clearly, $c_{\mathbf{q}}(\mathbf{u}) > 0$ if and only if $\mathbf{q} \in \mathcal{Q}_q(\mathbf{u})$.

Example 3.33. We have $c_{\texttt{"aa"}}(\texttt{"aaabaa"}) = 3$.

The so-called *q-gram profile* allows us to represent a string as a vector with integer elements of size $|\Sigma|^q$: $\mathcal{QP}_q(\mathbf{u}) = (c_{\mathbf{q}}(\mathbf{u}))_{\mathbf{q} \in \Sigma^q}$.

Example 3.34. Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$ and $q = 2$. We have:

$$\begin{array}{rccccccc} & & \texttt{"aa"} & \texttt{"ab"} & \texttt{"ba"} & \texttt{"bb"} & \\ \mathcal{QP}_q(\texttt{"abaa"}) = & (& 1, & 1, & 1, & 0 &), \\ \mathcal{QP}_q(\texttt{"aabbaa"}) = & (& 2, & 1, & 1, & 1 &). \end{array}$$

This leads us to the following dissimilarity measure proposed by Ukkonen in [453].

Definition 3.35. The *q-gram distance* is defined as:

$$\mathfrak{d}_{\mathrm{Q},q}(\mathbf{u}, \mathbf{v}) = \sum_{\mathbf{q} \in \Sigma^q} |c_{\mathbf{q}}(\mathbf{u}) - c_{\mathbf{q}}(\mathbf{v})|. \tag{3.14}$$

Note that the summation may be made just over $\mathbf{q} \in \mathcal{Q}_q(\mathbf{u}) \cup \mathcal{Q}_q(\mathbf{v})$.

Remark 3.36. According to [453, Theorem 2.1], $\mathfrak{d}_{\mathrm{Q},q}$ is a pseudometric for any q. It is not a metric as, for instance, for the bigram distance we have $\mathfrak{d}_{\mathrm{Q},2}(\texttt{"abaa"}, \texttt{"aaba"}) = 0$. It is because many strings may have the same q-gram profiles.

Ukkonen in [453] provides an $O(|\mathbf{u}|+|\mathbf{v}|)$-time and $O(|\Sigma|^q+|\mathbf{u}|+|\mathbf{v}|)$-space algorithm to compute the q-gram distance. Also please notice that Equation (3.14) is nothing more than the L_1 metric between $\mathcal{QP}_q(\mathbf{u})$ and $\mathcal{QP}_q(\mathbf{v})$ and thus the introduced dissimilarity measure can potentially be easily generalized.

C. Other string metrics

We should point out that many other string distances may be found in the literature. For instance, the Dinu rank distance $\mathfrak{d}_{\mathrm{DR}}$ [153], closely related to the so-called Spearman's footrule, see [149], has recently been of interest in computational biology. As for its construction we need some linear order on Σ now (in fact, its nature is irrelevant here), let us without loss of generality assume that $\Sigma = \{1, \ldots, k\} \subseteq \mathbb{N}$ – any set of characters may be mapped to a set of consecutive integers.

Let us define a mapping $\Sigma \ni u_i \mapsto (u_i, j_i) \in \mathbb{N}^2$, $i \in [d_u]$, $d_u = |\mathbf{u}|$, such that $j_i = \sum_{k=1}^{i} \mathbf{1}(u_k = u_i)$. In other words, e.g., $(3, 2)$ denotes the 2nd occurrence of character 3 in $\mathbf{u}$. From that we generate the sequence $\tilde{\mathbf{u}} = ((u_{\sigma_u(1)}, j_{\sigma_u(1)}), \ldots, (u_{\sigma_u(d_u)}, j_{\sigma_u(d_u)}))$, where σ_u stands for the ordering permutation of $\tilde{\mathbf{u}}$ with respect to the linear order $\preceq$ such that $(a, j) \preceq (a', j')$ if and only if either $a < a'$, or $a = a'$ and $j \leq j'$ (it is a lexicographic order on $\mathbb{N}^2$). Now for any $(a, j) \in \mathbb{N}^2$ let:

$$\mathrm{ord}_{\mathbf{u}}(a, j) = \begin{cases} \sigma_u(i) & \text{if } (a, j) = (\tilde{u}_i, \tilde{j}_i) \text{ for some } i, \\ 0 & \text{if } (a, j) \text{ is not a member of } \tilde{\mathbf{u}}. \end{cases}$$

Example 3.37. For instance, given a string $(2, 1, 1, 3, 3, 4, 1)$, we get:

i	u_i	j_i	$\tilde{u}_i$	$\tilde{j}_i$	$\mathrm{ord}_{\mathbf{u}}(\tilde{u}_i, \tilde{j}_i)$
1	2	1	1	1 *(first character 1)*	2
2	1	1	1	2 *(second character 1)*	3
3	1	2	1	3 *(third character 1)*	7
4	3	1	2	1 *(first character 2)*	1
5	3	2	3	1 *(first character 3)*	4
6	4	1	3	2 *(second character 3)*	5
7	1	3	4	1 *(first character 4)*	6

Definition 3.38. Let $\mathbf{u}, \mathbf{v} \in \Sigma^*$. The *Dinu rank distance* is given by:

$$\mathfrak{d}_{\mathrm{DR}}(\mathbf{u}, \mathbf{v}) = \sum_{(a,j) \in \tilde{\mathbf{u}} \cup \tilde{\mathbf{v}}} |\mathrm{ord}_{\mathbf{u}}(a, j) - \mathrm{ord}_{\mathbf{v}}(a, j)| \,. \tag{3.15}$$

Thus, it is an L_1 distance between the ord vectors. It may be shown that $\mathfrak{d}_{\mathrm{DR}}$ is a metric, see [153].

Figure A.20 gives our own implementation of the algorithm to compute the Dinu rank distance which operates in $O(d_u \log d_u + d_v \log d_v)$-time. Note that it is also possible to formulate it in such a way that it runs in $O(d_u|\Sigma| + d_v|\Sigma|)$-time. The costly step here is to find the stable ordering permutations of $\mathbf{u}$ and $\mathbf{v}$, but if more computations are needed on a set of strings (e.g., in implementations of hierarchical clustering algorithms that require roughly n^2 distance computations), they may be determined once in advance, and the time gets reduced to $O(d_u \vee d_v)$.

Other string (pseudo)metrics include, for example, the Jaro or Jaro-Winkler distance (see [471]), or the one proposed by Ehrenfeucht and Haussler in [181]. Note that the issue of how to compare DNA sequences is still in the top of a list of major open problems in bioinformatics/computational biology, see [476]. Yet, the discussed instances are perhaps the most frequently used and influential ones. Having said that, let us proceed with some seminal distance-based fusion function construction methods.

3.3.2 Median strings and a strings' centroid

The concept of a median string was introduced by Kohonen [294] for the purpose of smoothing of erroneous versions of strings and for string classification in, e.g., pattern recognition. Given $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in \Sigma^*$ it is a string $\mathbf{x}^*$ such that:

$$\mathbf{x}^* = \operatorname*{arg\,min}_{\mathbf{x}\in\Sigma^*} \sum_{i\in[n]} \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{x}),$$

where $\mathfrak{d}$ is some string metric, originally the Levenshtein distance. Additionally, we may consider a centroid-like search task:

$$\mathbf{x}^* = \operatorname*{arg\,min}_{\mathbf{x}\in\Sigma^*} \sum_{i\in[n]} \mathfrak{d}^2(\mathbf{x}^{(i)}, \mathbf{x}).$$

Note that many strings which are solutions to the two above equations may exist. Thus, the definition of a fusion function to aggregate a set of strings should be formulated with care.

Remark 3.39. Note that in the space of character strings, a medoid (set median) may in some contexts be more sensible than a median string, especially if n is large. This is especially the case when not all the strings in Σ^* are "valid" or "meaningful" (e.g., when we aggregate words in natural language). Recall that in the case of a medoid, we always get a string which is among those in the input data set. Another option is to restrict the search domain and seek within some *dictionary*.

The case of two strings. Let $n = 2$ and $\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \in \Sigma^*$. Both of the input objects are within the possible 1-median strings. In such a case, as a solution one may want to consider a string $\mathbf{x}$ such that $\mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{x}) = \lfloor \mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})/2 \rfloor$ and $\mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{x}) + \mathfrak{d}^2(\mathbf{x}^{(2)}, \mathbf{x}) = \mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})$ as a median string, which is exactly a solution to the string centroid problem. In other words, here a centroid is always at the same time a medoid.

To compute a Levenshtein metric-based centroid of two strings, we may make use of the fact that the classical algorithm that determines the value of this distance (see Equation (3.9)) also provides us with the information on how to edit the first string in such a way that the second one may be obtained. In order to do so, we can apply consecutive edit operations on the first string until the cumulative cost of edits made so far reaches $\lfloor \mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})/2 \rfloor$. This leads to an algorithm whose source code is given in Figure A.19. Note that the underlying fusion function is not symmetric. It can be made such if we first order the two input strings lexicographically.

Remark 3.40. The centroids of `"1234"` and `"2345"` with respect to the Levenshtein, LCS, and Damerau-Levenshtein distances are exactly `"234"` and `"12345"`. We observe that whichever we choose as a desired output of a centroid-like fusion function, the length internality property is violated.

However, in order to guarantee length internality, one may always restrict the search domain and be rather interested in finding, e.g.:

$$\mathbf{x}^* = \operatorname*{arg\,min}_{\mathbf{x} \in \bigcup_{d=d_{\min}}^{d_{\max}} \Sigma^d} \sum_{i \in [n]} \mathfrak{d}^p(\mathbf{x}^{(i)}, \mathbf{x}),$$

where $p \in \{1, 2\}$ and $d_{\min} = \bigwedge_{i=1}^{n} |\mathbf{x}^{(i)}|, d_{\max} = \bigvee_{i=1}^{n} |\mathbf{x}^{(i)}|$.

General case. There are string distances which guarantee that a median search is of polynomial-time. This is in the case of, e.g., the Dinu rank distance, see [156].

However, unlike in the fixed d case and the Hamming distance, it turns out that finding a median string with respect to the (weighted) Levenshtein distance is an NP-complete problem as a function of n even if Σ is a binary alphabet. Nicolas and Rivals in [374, 375] proved that by reduction to the intractable longest common subsequence problem. An exact algorithm was provided by Kruskal [301].

In order to find an approximate version of a median string with respect to the Levenshtein distance, Kohonen [294] suggests to compute the set median (which may be done easily) and then *to vary each of the symbol positions of the set median, making "errors" of all three types over the whole alphabet, and checking whether the sum of distances from the other elements is decreased.* More elaborate approximate algorithms, in the case of weighted Levenshtein distances, were given by Martinez et al. [350] (together with an application in k-nearest neighbor classification) and Abreu and Juan in [2] – yet, they are

also based on perturbations over the initial string. On the other hand, Jiang et al. [262] incorporate an idea of computing median strings by embedding them into Euclidean vector spaces. See also the works by Kohonen and Somervuo [285, 433] for an application in constructing unsupervised self-organizing maps (SOMs).

Here we shall provide a genetic algorithm (see Algorithm 2.72) to compute the desired fusion function. Its most interesting facet concerns a proper crossover and mutation scheme, the selection of which might not be trivial in the space of vectors of arbitrary lengths. We recommend the following approaches:

— a *crossover* between $\mathbf{u}$ and $\mathbf{v}$ is set to be the centroid of $\{\mathbf{u}, \mathbf{v}\}$ (see above),

— a single *mutation* operation may consist of a Levenshtein edit: with equal probability we choose to perform at a randomly chosen index in a string being mutated, either:

  – an insertion of a character,
  – a removal of a character, or
  – a replacement of a character with one sampled from Σ.

Remark 3.41. The discussed crossover scheme is sometimes called an *intermediate recombination*. We observe that a cut-and-splice crossover (joining a random prefix of $\mathbf{u}$ with a random suffix of $\mathbf{v}$) does not perform well. Also what does not work best is a scheme used in [155] (for center strings with respect to the Dinu rank distance, see below), which basically is based on joining a random prefix of the first vector with a permuted version of a sampled suffix of the second vector.

Remark 3.42. Note that finding the median with respect to the q-gram distance is much more difficult.

Let $P = \{\mathbf{q}^{(1)}, \ldots, \mathbf{q}^{(m)}\}$ be a set of all q-grams that appear in at least one of the input strings, i.e., $P = \bigcup_{i=1}^{n} \mathcal{Q}_q(\mathbf{x}^{(i)})$. We remap each string $\mathbf{x}^{(i)}$ to a q-gram profile $\mathbf{c}^{(i)} = (c_{\mathbf{q}^{(1)}}(\mathbf{x}^{(i)}), \ldots, c_{\mathbf{q}^{(m)}}(\mathbf{x}^{(i)}))$, which gives a vector of nonnegative integers.

Our aim is to find:

$$\arg\min_{\mathbf{c} \in X} \sum_{i=1}^{n} \mathfrak{d}_1(\mathbf{c}, \mathbf{c}^{(i)}), \tag{3.16}$$

where X is a subset of $\mathbb{N}_0^m$ which denotes a valid q-gram profile, i.e., one from which we may reconstruct a proper character string.

If the search space was just as simple as $\mathbb{N}_0^m$, an integer programming (IP) solver could be used for determining the 1-median (perhaps one that is able to iterate through all the optimal solutions). Yet, for instance, assume that `"abcb"`, `"cbac"`, `"acab"` are three input character strings and $q = 2$. Then the output from an IP solver suggests that the best match consists of the following bigrams: `"ac"`, `"cb"`, `"ab"`. It is easily seen that no string can be constructed from such a q-gram (multi)set.

3.3.3 Closest strings

Recall that the 1-center problem aims at finding a point which minimizes the maximal distance towards every point in a given input data set. What is known in the literature under the name *closest* or *center* string problem represents exactly such a type of task, this time however – in the character string domain.

More precisely, let $\mathfrak{d}$ be a (pseudo)metric on Σ^*. Given $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, we define:

$$\mathsf{ClosestString}_{\mathfrak{d}}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \operatorname*{arg\,min}_{\mathbf{x} \in \Sigma^*} \bigvee_{i \in [n]} \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{x}).$$

Note that the solution may be ambiguous. There are many applications of such a fusion function in computational biology. Among some instances listed in [155] we find: searching for motifs or common patterns in a set of given DNA sequences or genetic drugs design with a structure similar to a set of RNA sequences.

Remark 3.43. For $n = 2$ a centroid of $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\}$ is also its center string.

If $\mathfrak{d}$ is the ordinary Levenshtein distance and $|\Sigma| \geq 2$, then – as shown by Nicolas and Rivals in [374] – the center string is NP-complete with respect to n (a proof is by reduction to the longest common subsequence problem). Moreover, the mentioned authors show similar results for the case of the weighted Levenshtein distance in [375].

Here, also an algorithm for finding the closest string with respect to the Dinu rank distance is NP-complete, see [157]. Thus, in [155] Dinu and Ionescu propose a genetic algorithm-based approach to approximate a closest string. Moreover, in [154] they develop k-means-like and hierarchical clustering methods based on closest strings and the rank distance.

Chapter 4

Aggregation of other data types

Fusion functions defined on more complex domains than in the previous chapters are the subject of interest in this part of the monograph. In the consecutive sections we shall assume that we deal with the following data types:

1. directional (e.g., angular) data,
2. real intervals,
3. fuzzy numbers,
4. random variables,
5. trees and other graphs as well as rankings and other relations,
6. general finite semimetric spaces,
7. heterogeneous data sets.

We already observed that it is possible to aggregate fusion functions (in particular, regression and classification models) and metrics themselves. Even though the construction and analysis of data fusion tools acting on the aforementioned domains may seem much more difficult, it shall turn out that many of the already known ideas may be easily extrapolated. For instance, if we deal with a linear space (and thus if we are able to define addition and scalar multiplication operations properly), then a weighted arithmetic mean, i.e., a convex combination, can be defined. If a linear ordering relation may be introduced quite naturally, then we can refer to the notion of an order statistic. Moreover, having various metrics or other kinds of dissimilarity measures, we may consider the notion of a penalty-based fusion function.

4.1 Directional data

Let us consider the situation where observations are defined on spheres $\{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| = 1\} = \mathbb{S}^{d-1} \subset \mathbb{R}^d$, $d > 1$, rather than Euclidean spaces like in Chapter 2, see [190, 332, 333]. For instance, this kind of information may occur in:

— data on location of earthquake epicenters (as Earth may be modeled by a sphere),

— observations of winds, animal migration, paleomagnetism, etc. (here, a natural phenomenon's movement direction is the most relevant),

— events occurring periodically, e.g., on a 24-hour clock, yearly calendar (whenever there is a cyclic pattern in time),

— handwriting features descriptions (for, e.g., optical character recognition, see [20]),

— models of local protein structure, see [67],

and many others.

Handling *directional* (e.g, circular/angular for $d = 2$ or spherical for $d = 3$) data is quite challenging. Even if we are on a circle, we do not have a natural ordering of our data. This is because angles of $-180°$ and $180°$ are equivalent, as well as $0°$ and $360°$, and so on. Additionally, observe that the "average" of $165°$ and $-165°$ should not be set to $0°$, etc. Our space of discourse here may be conceived as a "modulo 2π"-type algebra.

Example 4.1. A *rose diagram* is a modification of an ordinary histogram, tailored for depicting circular data. Figure 4.1 depicts a rose diagram of an exemplary circular data set, being a random sample from a von Mises distribution (the circular counterpart of a normal distribution) with expected value of $\pi/4$.

Remark 4.2. What should be the mean of $0, \pi/2, \pi, 3\pi/2$? Of course, there is no definite answer to this question.

Let us consider a few fusion functions that may be found in the literature and which aim to provide information on the *average* value of a directional data set. In other words, for $d > 1$ and some n, this time we are interested in fusion functions like $\mathsf{F} : (\mathbb{S}^{d-1})^n \to \mathbb{S}^{d-1}$.

Firstly, we should note that in the case of directional data, most researchers discourage using a stereographic projection of input data (see [432]), i.e., embedding $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$ in $\mathbb{R}^d$. Instead, using the Euclidean space analogues are preferred. Therefore, one may consider:

— arcs of a great circle as replacements for straight lines (the "shortest curve" joining two points),

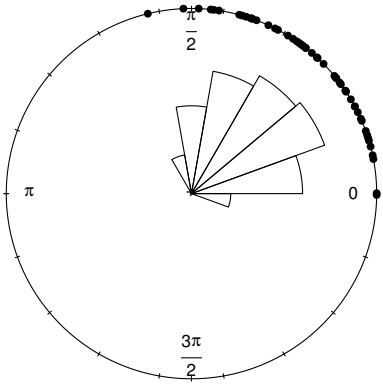


Figure 4.1. An exemplary rose diagram of a circular data set.

— arc lengths as replacements for the Euclidean distance,

and so on.

Example 4.3. The mean of a circular data set $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)} \in \mathbb{S}^1$ is usually defined as follows, see [259]. Let $\vartheta_1, \ldots, \vartheta_n \in [-\pi, \pi[$ denote the corresponding angles. Then:

$$\begin{aligned} & \mathsf{CircMean}(\boldsymbol{\vartheta}) \\ = \; & \operatorname{atan2}\Big(\mathsf{AMean}(\sin\vartheta_1, \ldots, \sin\vartheta_n), \mathsf{AMean}(\cos\vartheta_1, \ldots, \cos\vartheta_n)\Big), \end{aligned}$$

where for $x, y \in \mathbb{R} \setminus \{0\}$:

$$\operatorname{atan2}(y, x) = \begin{cases} \arctan\frac{y}{x} & \text{if } x > 0, \\ \arctan\frac{y}{x} + \pi & \text{if } x \leq 0 \text{ and } y > 0, \\ \arctan\frac{y}{x} - \pi & \text{if } x \leq 0 \text{ and } y \leq 0. \end{cases}$$

Example 4.4. The Mardia-type (see [333]) median of a circular data set is a point $\mathbf{y}$ on the unit circle such that:

— most of the observations are closer to the median $\mathbf{y}$ than to the anti-median $\mathbf{y}'$,

— the number of observations in each semi-circle determined by the diameter $\mathbf{y}\mathbf{y}'$ are equal.

Note that for some data sets the Mardia-type median may be ambiguous, see [381] for discussion and some modifications of the above method.

Example 4.5. The extension of the 1-median is called *the Mardia-Fisher spherical median*. It is a point $\mathbf{y}$ such that:

$$\mathbf{y} = \arg\min_{\mathbf{y}\in\mathbb{S}^{d-1}} \sum_{i=1}^{n} \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{y}),$$

where $\mathfrak{d}(\mathbf{x}, \mathbf{y}) = \cos^{-1}(\langle \mathbf{x}, \mathbf{y}\rangle)$, i.e., the length of the geodesic arc joining $\mathbf{x}$ and $\mathbf{y}$.

Example 4.6. Data depth notions were also generalized for the case of directional data. For instance, an Oja-type spherical median was proposed in [431] – instead of simplices we consider the intersection of all closed hemispheres that contain $d + 1$ points. Moreover, in [325] the concept of angular simplical depth and angular Tukey's depth (rotation invariant) is considered and in [318] – angular Mahalanobis depth.

4.2 Aggregation of real intervals

If input data of numeric type are not precisely given, they sometimes are represented as real intervals, see, e.g., [24, 281, 302]. Let $\mathcal{I}([a, b])$ denote the set of all closed subintervals of $[a, b]$, $a < b$.

Example 4.7. In statistics, data may be provided by means of frequency tables (which may be much easier to gather manually or using low accuracy measurement devices). An exemplary grouped (histogram-like) data set is as follows:

time	**frequency**
$[0, 5[$	5
$[5, 10[$	9
$[10, 15[$	6
$[15, 20[$	3

Example 4.8. Recall from page 30 that most values $x \in \mathbb{R}$ cannot be directly represented in a set of floating-point numbers, $\mathbb{F}$. Instead of simple rounding, we may model x as the smallest interval $[\underline{x}, \overline{x}]$, where $\underline{x}, \overline{x} \in \mathbb{F}$ and $x \in [\underline{x}, \overline{x}]$. Surely, $\underline{x} = \underline{\mathrm{fp}}(x) \leq x$ is defined as x rounded towards $-\infty$ and $\overline{x} = \overline{\mathrm{fp}}(x) \geq x$ – towards $+\infty$. For instance, the GNU C library allows (on CPU architectures and compilers that support this operation) to change the rounding mode by a call to the `int fesetround(int round)` function, where `round` $\in$ {`FE_TONEAREST`, `FE_UPWARD`, `FE_DOWNWARD`, `FE_TOWARDZERO`}.

In the current setting, a few approaches to interval data fusion are possible.

Intervals as bounded posets. On the set of real intervals we may define a partial ordering relation, e.g., as follows:

— $[\underline{x},\overline{x}] \sqsubseteq_I [\underline{y},\overline{y}]$ if and only if $\overline{x} < \underline{y}$, or $\underline{x} = \underline{y}$ and $\overline{x} = \overline{y}$ (the so-called *interval order*),

— $[\underline{x},\overline{x}] \leq^2 [\underline{y},\overline{y}]$ if and only if $\underline{x} \leq \underline{y}$ and $\overline{x} \leq \overline{y}$ (Cartesian product extension of ordinary $\leq$ on the set of real numbers).

In both cases we may obtain a bounded lattice, thus aggregation methods developed already in Section 1.7 are directly applicable here, see also [145]. Of course, it is possible to derive more tailored results as well, such as ones concerning for instance t-norms on the space of real intervals and the $\leq^2$ order, see [138, 491]. In particular, it can be shown that for every semicontinuous t-norm T on $\mathcal{I}([0,1])$ there exists a t-norm T' on $[0,1]$ such that $\mathsf{T}([\underline{x},\overline{x}],[\underline{y},\overline{y}]) = [\mathsf{T}'(\underline{x},\underline{y}),\mathsf{T}'(\overline{x},\overline{y})]$. Moreover, Làzaro and Calvo in [307] considered aggregation functions monotone with respect to the above orders.

Defuzzification. Note that each interval $[\underline{x},\overline{x}]$ may be represented as $x \pm r$, where $x = (\underline{x}+\overline{x})/2$ is its midpoint and $r = (\overline{x}-\underline{x})/2 \geq 0$ is its halfwidth. Some statistical data analysis handbooks suggest to "defuzzify" interval data and instead just to consider the corresponding midpoints (corresponding halfwidths may be aggregated separately to measure the imprecision of the outcome). Then, classical fusion functions for unidimensional quantitative data may be used.

Interval arithmetic. Let us introduce the following extensions of arithmetic operations to the space of real intervals, see, e.g., [281]:

$$\begin{aligned}
\big[\underline{x},\overline{x}\big] \oplus \big[\underline{y},\overline{y}\big] &= \big[\underline{x}+\underline{y}, \overline{x}+\overline{y}\big],\\
\big[\underline{x},\overline{x}\big] \ominus \big[\underline{y},\overline{y}\big] &= \big[\underline{x}-\overline{y}, \overline{x}-\underline{y}\big],\\
\big[\underline{x},\overline{x}\big] \otimes \big[\underline{y},\overline{y}\big] &= \big[\underline{x}\cdot\underline{y} \wedge \overline{x}\cdot\overline{y} \wedge \underline{x}\cdot\overline{y} \wedge \overline{x}\cdot\underline{y}, \underline{x}\cdot\underline{y} \vee \overline{x}\cdot\overline{y} \vee \underline{x}\cdot\overline{y} \vee \overline{x}\cdot\underline{y}\big],\\
\big[\underline{x},\overline{x}\big] \oslash \big[\underline{y},\overline{y}\big] &= \big[\underline{x},\overline{x}\big] \otimes \big[1/\overline{y}, 1/\underline{y}\big] \text{ whenever } 0 \notin \big[\underline{y},\overline{y}\big].
\end{aligned}$$

Functions like $\mathsf{f} : \mathbb{R} \to \mathbb{R}$ may be extended straightforwardly. For example, if f is strictly increasing, then let:

$$\textcircled{\mathsf{f}}([\underline{x},\overline{x}]) = [\mathsf{f}(\underline{x}),\mathsf{f}(\overline{x})].$$

Remark 4.9. The Interval Arithmetic Library in Boost for the C++ programming language [83] is able to programmatically quantify the propagation of rounding errors in floating point computations by using proper rounding towards $-\infty$ (left) and $+\infty$ (right bound).

Note that for any $s \geq 0$ it holds $\big[\underline{x},\overline{x}\big] \otimes s = \big[\underline{x},\overline{x}\big] \otimes [s,s] = \big[s\underline{x}, s\overline{x}\big]$. We see that the set of intervals is closed under addition and scalar multiplication and forms a linear space. Thus, the notion of a weighted arithmetic mean may

easily be introduced. This leads to an idempotent and $\leq^2$-monotone fusion function. On the other hand, redefining OWA-like operators is not as trivial, as the construction of a linear order on $\mathcal{I}([a,b])$ can be done in many ways (e.g., by considering intervals' midpoints, halfwidths, etc.).

Penalty-based fusion functions. In order to introduce penalty-based functions to aggregate interval data, let us first recall the most popular interval metrics, see also [39, Chapter 8].

Definition 4.10. *Moore's interval metric* is given by:

$$\mathfrak{d}_{\mathrm{M}}\left(\left[\underline{x},\overline{x}\right] \oplus \left[\underline{y},\overline{y}\right]\right) = |\underline{x}-\underline{y}| \vee |\overline{x}-\overline{y}|. \tag{4.1}$$

If $[a,b]$ is interpreted as a point in $\mathbb{R}^2$, the Moore metric is exactly the Chebyshev distance, $\mathfrak{d}_\infty$. On the other hand, if we rely on the midpoint $\pm$ halfwidth representation, then this metric is the $\mathfrak{d}_1$ one: it holds $\mathfrak{d}_{\mathrm{M}}(x \pm r_x, y \pm r_y) = |x-y| + |r_x - r_y|$. Therefore, we have what follows (compare also Section 2.5).

— The $\mathfrak{d}_{\mathrm{M}}$-based 1-median of $\mathbf{x} \in \mathcal{I}([a,b])^n$ is equal to the componentwise median of the inputs' midpoints and halfwidths, see [114, Theorem 1].

— The $\mathfrak{d}_{\mathrm{M}}$-based 1-center of $\mathbf{x} \in \mathcal{I}([a,b])^n$ is given as:

$$\Big[(\bigvee_i \underline{x}_i + \bigwedge_i \underline{x}_i)/2, (\bigvee_i \overline{x}_i + \bigwedge_i \overline{x}_i)/2\Big], \tag{4.2}$$

see [114, Theorem 2],

Moreover, if we assume that $\mathfrak{d}_{\mathrm{M}_2}(x \pm r_x, y \pm r_y) = \sqrt{|x-y|^2 + |r_x - r_y|^2}$, then:

— $\mathfrak{d}_{\mathrm{M}_2}$-based centroid of $\mathbf{x} \in \mathcal{I}([a,b])^n$ is equal to the componentwise arithmetic mean of midpoints and halfwidths, see [114, Theorem 3].

Note that the above results may easily be generalized to hyperrectangles such that their faces are parallel to axes of a coordinate system. Such data occur, among others, in the so-called granular [24, 386] box regression, see [387] and also [237, 388].

Among other interval metrics we find the Wasserstein one, $\mathfrak{d}_{\mathrm{W}}(x \pm r_x, y \pm r_y) = \sqrt{(x-y)^2 + (r_x - r_y)/3}$ and the Bertoluzza one $\mathfrak{d}_{\mathrm{B}}(x \pm r_x, y \pm r_y) = \sqrt{(x-y)^2 + 2(r_x - r_y)/3}$, see, e.g., [257] for a review and a possible application in data clustering.

4.3 Aggregation of fuzzy numbers

Fuzzy set theory lets us to quite intuitively represent imprecise or vague information, see [281]. Fuzzy numbers (FNs), introduced by Dubois and Prade in

[164], form a particular subclass of fuzzy sets of the real line. They play an important role in many practical applications, e.g., in automation and robotics [271], statistical process control [251], survey design in the social sciences [140] or decision making [116], since we often describe our knowledge about objects through vague numbers such as "I'm about 180 cm tall" or "The train arrived between 2 and 3 p.m.".

Definition 4.11. A fuzzy set A with membership function $\mu_A : \mathbb{R} \to [0,1]$ is a *fuzzy number*, if it possesses at least the four following properties:

(a) it is a normalized fuzzy set, i.e., $\mu_A(x_0) = 1$ for some $x_0 \in \mathbb{R}$,

(b) it is fuzzy convex, i.e., for any $x_1, x_2 \in \mathbb{R}$ and $\lambda \in [0,1]$ it holds:

$$\mu_A(\lambda x_1 + (1-\lambda)x_2) \geq \mu_A(x_1) \wedge \mu_A(x_2),$$

(c) the support of A is bounded, where:

$$\text{supp}(A) = \text{cl}(\{x \in \mathbb{R} : \mu_A(x) > 0\}),$$

(d) μ_A is upper semicontinuous.

Remark 4.12. It may be shown that the membership function of a fuzzy number A is given by:

$$\mu_A(x) = \begin{cases} 0 & \text{if} \quad x < a_1, \\ l_A(x) & \text{if} \quad a_1 \leq x < a_2, \\ 1 & \text{if} \quad a_2 \leq x \leq a_3, \\ r_A(x) & \text{if} \quad a_3 < x \leq a_4, \\ 0 & \text{if} \quad x > a_4, \end{cases} \tag{4.3}$$

where $a_1, a_2, a_3, a_4 \in \mathbb{R}$, $l_A : [a_1, a_2] \longrightarrow [0,1]$ is a nondecreasing upper semicontinuous function, $l_A(a_1) = 0$, $l_A(a_2) = 1$, called the *left side* of the fuzzy number, and $r_A : [a_3, a_4] \longrightarrow [0,1]$ is a nonincreasing upper semicontinuous function, $r_A(a_3) = 1$, $r_A(a_4) = 0$, called the *right side* of the fuzzy number A.

Remark 4.13. A fuzzy number A may also be specified by providing its so-called α-cuts, $\alpha \in [0,1]$. Let:

$$A_\alpha = \{x \in \mathbb{R} : \mu_A(x) \geq \alpha\} \tag{4.4}$$

for $\alpha > 0$ and $A_0 = \text{supp}(A)$. The 1-cut is sometimes called the core of A. Every α-cut is a closed interval, that is:

$$A_\alpha = [A_L(\alpha), A_R(\alpha)], \tag{4.5}$$

with:

$$A_L(\alpha) \quad = \quad \inf\{x \in \mathbb{R} : \mu_A(x) \geq \alpha\},$$

$$A_R(\alpha) \quad = \quad \sup\{x \in \mathbb{R} : \mu_A(x) \geq \alpha\}.$$

Note that if the sides of the fuzzy number A are strictly monotone, then A_L and A_U are inverse functions of l_A and r_A, respectively.

Let $\mathbb{F}(\mathbb{R})$ denote the set of all fuzzy numbers. In practice, e.g., when computations of arithmetic operations are performed, fuzzy numbers with simple membership functions are often preferred. A very useful subclass of $\mathbb{F}(\mathbb{R})$, especially for computer processing, may be defined by considering fuzzy numbers with piecewise linear side functions. Thus, let us consider the following definition, see [128, 129].

Definition 4.14. Fix $n \in \mathbb{N}_0$. Given $\boldsymbol{\alpha} \in \{(\alpha_0, \alpha_1, \ldots, \alpha_{n+1}) \in [0,1]^{n+2} : 0 = \alpha_0 < \alpha_1 < \cdots < \alpha_n < \alpha_{n+1} = 1\}$ and $\mathbf{s} \in \{(s_1, \ldots, s_{2n+4}) \in \mathbb{R}^{2n+4} : s_1 \leq \cdots \leq s_{2n+4}\}$, an *$\boldsymbol{\alpha}$-piecewise linear n-knot fuzzy number* $S(\boldsymbol{\alpha}, \mathbf{s})$, is defined by:

$$\begin{aligned} S(\boldsymbol{\alpha}, \mathbf{s})_L(\beta) &= s_{i+1} + (s_{i+2} - s_{i+1}) \frac{\beta - \alpha_i}{\alpha_{i+1} - \alpha_i}, \\ S(\boldsymbol{\alpha}, \mathbf{s})_U(\beta) &= s_{2n+4-i} + (s_{2n+3-i} - s_{2n+4-i}) \frac{\beta - \alpha_i}{\alpha_{i+1} - \alpha_i}, \end{aligned}$$

for some $i \in [0:n]$ such that $\beta \in [\alpha_i, \alpha_{i+1}]$.

Please note that the membership function of $S(\boldsymbol{\alpha}, \mathbf{s})$ is also piecewise linear in the case when $\mathbf{s}$ is strictly monotone (for an example see Figure 4.2).

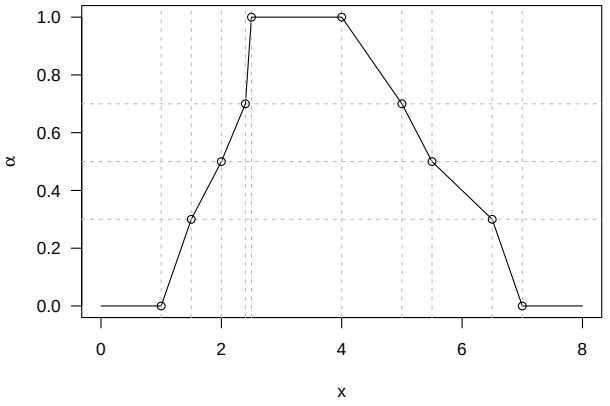


Figure 4.2. Plot of an exemplary 3-knot piecewise linear fuzzy number $S(\boldsymbol{\alpha}, \mathbf{s})$, with $\boldsymbol{\alpha} = (0, 0.3, 0.5, 0.7, 1)$ and $\mathbf{s} = (1, 1.5, 2, 2.4, 2.5, 4, 5, 5.5, 6.5, 7)$.

Remark 4.15. It is worth noting that the class of fuzzy sets introduced in Definition 4.14 generalizes some well-known subfamilies of fuzzy numbers. Actually, for $n = 0$ and $s_1 = s_4$ we get "crisp" real numbers, for $n = 0$ and $s_1 = s_2, s_3 = s_4$ we obtain "crisp" real intervals; if $n = 0$ and $s_2 = s_3$ we get triangular fuzzy numbers, and by assuming only $n = 0$ we obtain trapezoidal fuzzy numbers.

Further on we assume that two fuzzy numbers A and B are equal (denoted $A = B$) if $A_L(\alpha) = B_L(\alpha)$ and $A_U(\alpha) = B_U(\alpha)$ for all $\alpha \in [0, 1]$.

Let us consider a fusion function $\mathsf{F} : \mathbb{F}(\mathbb{R})^n \to \mathbb{F}(\mathbb{R})$, which aims to aggregate n fuzzy numbers so that one fuzzy number is generated as a result.

Defuzzification methods. Concepts such as the expected value [167] or value [144] of a fuzzy number (see Section 5.6) may be used to defuzzify a given fuzzy number. Together with some measure of nonspecifity, e.g., width [112] or ambiguity [144], these may be used to concisely represent $A \in \mathbb{F}(\mathbb{R})$ as $x \pm r$, i.e., a real interval. If such a level of data loss is accepted, then the aggregation methods discussed in the previous section may be utilized.

Arithmetic operations. In order to extend a binary arithmetic operation $*$ (e.g., $+$, $-$, $\times$, $/$) to the set of fuzzy numbers, most often Zadeh's *extension principle*, see [270, 281], is used. In such a case $C = A \circledast B$ is given via the membership function:

$$\mu_C(z) = \sup_{z=x*y} \left(\mu_A(x) \wedge \mu_B(y)\right). \tag{4.6}$$

Note that for fuzzy numbers being real intervals, the extension principle generates exactly the same arithmetic operators as presented in the previous section. In particular, in the α-cut representation, the sum $A \oplus B$ and the scalar multiplication $t \otimes A$ (see, e.g., [151, page 40]) for every $\alpha \in [0, 1]$ is given by:

$$(A \oplus B)_\alpha = A_\alpha \oplus B_\alpha = \left[A_L(\alpha) + B_L(\alpha), A_U(\alpha) + B_U(\alpha)\right] \tag{4.7}$$

and:

$$(t \otimes A)_\alpha = t \otimes A_\alpha = \begin{cases} \left[t \cdot A_L(\alpha), t \cdot A_U(\alpha)\right] & \text{if } t \geq 0, \\ \left[t \cdot A_U(\alpha), t \cdot A_L(\alpha)\right] & \text{if } t < 0. \end{cases} \tag{4.8}$$

Remark 4.16. Note that a set of piecewise linear fuzzy numbers with fixed knot configuration is closed under addition and scalar multiplication, but not, e.g., multiplication of two arbitrary members of this class. Nevertheless, this suffices to introduce the notion of a weighted arithmetic mean.

Basic fuzzy number arithmetic operations are available in R via the FuzzyNumbers package [210]. For practical reasons, each arbitrary fuzzy number should be approximated by a piecewise linear one (using a considerable number of knots), see [129] for discussion.

```
library("FuzzyNumbers")
A <- TrapezoidalFuzzyNumber(1, 2, 3, 4)
B <- TriangularFuzzyNumber(3, 5.5, 6)
C <- as.PiecewiseLinearFuzzyNumber(A, knot.n=100) *
     as.PiecewiseLinearFuzzyNumber(B, knot.n=100)
alphacut(C, c(0, 1)) # support and core
##        L      U
## 0      3   24.0
## 1     11   16.5
```

Please note that the extension principle is based on the $\wedge$ operation. It turns out that this way of extending arithmetic operations to the set of fuzzy numbers can be generalized by replacing $\wedge$ in Equation (4.6) with, e.g., an arbitrary triangular norm. This leads to the notion of the so-called interactive fuzzy numbers, where one is able to take into account a kind of mutual interdependency between such types of objects (compare the role of copulas in probability theory). This idea has been investigated by Fullér and other researchers, see, e.g., [102, 126, 127, 163, 203].

Orders in the space of fuzzy numbers. The space of fuzzy numbers, just like its subclass – real intervals, has no natural linear order. A relation $A \sqsubseteq_S B$ whenever, e.g., $\sup \operatorname{supp} A \leq \inf \operatorname{supp} B$ or $A = B$ is merely a partial ordering.

Nevertheless, in the literature many authors have considered different ways to construct so-called ranking indices, i.e., functions of the $r : \mathbb{F}(\mathbb{R}) \to \mathbb{R}$ kind, which can be used to construct a total preorder on the set of fuzzy numbers. Such tools may be useful for symmetrizing weighted arithmetic means in order to define OWA-like operations.

In particular, Ban and Coroianu in [21] characterized all the ranking indices for trapezoidal fuzzy numbers that fulfill – among others – the set of famous Wang and Kerre [462] axioms, including translation and scale invariance. Denoting a trapezoidal fuzzy number as $\mathrm{T}(s_1, s_2, s_3, s_4)$, this very strong result indicates that the only reasonable ranking index may be a kind of linear combination of $s_1, \ldots, s_4$, given by:

$$r(\mathrm{T}) = cs_1 + (0.5 - c)s_2 + (0.5 - c)s_3 + cs_4$$

for some $c \in [0, 1]$.

Metrics in the space of fuzzy numbers and penalty-based fusion functions. Perhaps the most often considered metric in the space of fuzzy numbers is an extension of the Euclidean distance defined by the equation:

$$\mathfrak{d}_2(A, B) = \sqrt{\int_0^1 \left(A_L(\alpha) - B_L(\alpha)\right)^2 d\alpha + \int_0^1 \left(A_U(\alpha) - B_U(\alpha)\right)^2 d\alpha}. \quad (4.9)$$

In a very similar manner, arbitrary weighted Minkowski distances may be introduced, see [235].

Ban, Coroianu, and Grzegorzewski in [22] considered a trapezoidal fuzzy number fusion problem. They derived an algorithm for determining a $\mathfrak{d}_2$-based centroid of n such fuzzy sets using the Karush-Kuhn-Tucker theorem, which is expressed as:

$$\mathsf{F}(\mathbf{t}^{(1)}, \ldots, \mathbf{t}^{(n)}) = \mathrm{T}\left(\mathsf{AMean}(s_1^{(1)}, \ldots, s_1^{(n)}), \ldots, \mathsf{AMean}(s_4^{(1)}, \ldots, s_4^{(n)})\right),$$

where $\mathbf{t}^{(i)} = \mathrm{T}(s_1^{(i)}, \ldots, s_4^{(i)})$.

Additionally, the same authors in [23] studied the conditions for which, given a metric $\mathfrak{d}$ in the space of fuzzy numbers, the corresponding 1-median exists and is unique. It is worth noting that their results are based on the Rådström embedding theorem and can be quite easily generalized to some other linear spaces equipped with a norm-generated metric.

Some notes on aggregation of other types of fuzzy quantities. There are various possible ways to generalize and/or extend the theory of classical fuzzy sets. One of them includes the class of Atanassov's so-called intuitionistic fuzzy sets (AIFS, see, e.g., [16, 17]). Here, the degrees of "belongingness" and "nonbelongingness" of an observation to an AIFS are modeled separately. Notably, AIFS are equivalent to interval-valued fuzzy sets (see [147] for discussion), so we may rather just model the degree of belongingness using a real interval.

In particular, e.g., Szmidt and Kacprzyk in [439] as well as Grzegorzewski in [236] review possible ways to define metrics in the space of AIFS. Moreover, Deschrijver in [146] defines OWA operators together with quasi-arithmetic means, and Beliakov, Bustince, James, Calvo, and Fernandez [40] define median-like fusion functions. The reader is referred to [39] for a comprehensive review of these concepts and a list of practical applications of AIFS, e.g., in image processing.

4.4 Aggregation of random variables

Let us assume that we are given n random variables which are independent and identically distributed (i.i.d.) – following a common cumulative distribution function F. That is, let $\mathbf{X} = (X_1, \ldots, X_n)$ i.i.d. F. Moreover, let F be continuous with support $\mathbb{I} = [a, b]$. Note that basically in aggregation theory we only consider observed values such as $\mathbf{x} = (x_1, \ldots, x_n)$, i.e., particular realizations of $\mathbf{X}$. Probabilistic models provide us with yet another way for dealing with input data imprecision (in fact, one that is accepted by most of the practitioners).

Definition 4.17. A *statistic* is any function of random variables.

Thus, each fusion function defined on a sequence of random variables is a statistic in the probabilistic sense. Note that $F(X_1, \ldots, X_n) = Y$ is per se another random variable which follows its own distribution function.

Studies of probabilistic properties of particular classes of fusion functions appear significantly less frequently in the literature than research dealing with

constructions of particular functions fulfilling desired statistical properties. Despite this, let us now review a few fundamental results on general properties of some fusion functions discussed so far.

Order statistics. Here are some basic facts on order statistics in an i.i.d. model, see [135, Chapter 2]. The cumulative distribution function of the ith order statistic is given by:

$$F_{(i)}(x) = \sum_{j=i}^{n} \binom{n}{j} F^j(x)(1-F(x))^{n-j} = I_{F(x)}(i, n-i+1) \tag{4.10}$$

and – assuming that f is the common density of each X_i – the probability density function is given by:

$$f_{(i)}(x) = \frac{F^{i-1}(x)(1-F(x))^{n-j} f(x)}{B(i, n-i+1)}, \tag{4.11}$$

where $B(x,y) = \int_0^1 t^{x-1}(1-t)^{y-1}\,dt$ is the Beta function and $I_p(x,y) = \int_0^p t^{x-1}(1-t)^{y-1}\,dt/B(x,y)$ is the regularized incomplete Beta function, $x, y > 0$, $p \in [0,1]$.

In particular, the sample median for even n follows the c.d.f.:

$$F_{\mathsf{Median}}(x) = \frac{2}{B(n/2, n/2)} \cdot$$
$$\cdot \int_{-\infty}^{x} F(y)^{0.5n-1} \left((1-F(y))^{0.5n} - (1-F(2x-y))^{0.5n}\right) f(y)\,dy,$$

see [148] for a proof.

Example 4.18. The ith order statistic of a sample of i.i.d. random variables uniformly distributed on $[0,1]$ has a Beta distribution with parameters i and $n+1-i$. In any case, generally we can observe that deriving exact yet user-friendly forms of order statistics' distributions is a difficult task.

We already mentioned that a statistic is a random variable itself. For large n and arbitrary $p \in]0,1]$, the $\lceil np \rceil$th order statistic is approximately normally distributed. More precisely:

$$X_{(\lceil np \rceil)} \sim \mathrm{AN}\left(\mathsf{F}^{-1}(p), \frac{\sqrt{p(1-p)}}{\sqrt{n} f(F^{-1}(p))}\right),$$

where $\mathrm{AN}(\mu, \sigma)$ denotes that the distribution is asymptotically normal with expected value μ and standard deviation σ.

Of course, do notice that $X_{(i)}$ and $X_{(j)}$ are no more independent random variables. However, in the literature we may find general equations giving joint distributions of pairs, triples, etc., of order statistics. For more details on stochastic properties of order statistics and functions of order statistics the reader is referred to the seminal monograph by David and Nagaraja [135].

Remark 4.19. Provided that F is symmetric, the sample median is one of the possible *estimators* of the expected value of X, that is – intuitively – fusion functions which aim to *guess* one of the parameters or characteristics of the unknown probability distribution – solely based on the observed sample. What is worth noting, "good statistical properties", such as unbiasedness, consistency, efficiency, and so forth, may suggest which fusion function shall be chosen for use in particular applications (see Section 5.1). For instance, it is known that the arithmetic mean is an unbiased, minimal variance estimator of the expected value provided that X_i has finite variance, which in simple statistical models (e.g., not contaminated by outliers) is a much better choice than the median.

Weighted arithmetic means and ordered weighted averages. By the famous central limit theorem we know that the arithmetic mean is asymptotically normally distributed (under certain conditions on F). For arbitrary weighted means, if the expected value of X is finite, then the expected value of a weighted mean is equal to the expected value of X, because $\mathbb{E}\,\mathsf{WMean}_{\mathbf{w}}(\mathbf{X}) = \sum_{i=1}^{n} w_i \mathbb{E}\,X_i = \mathbb{E}\,X$.

Interestingly, in probability and statistics, OWA operators are special cases of the so-called L-statistics, i.e., linear combinations of order statistics. Their properties are quite well-known already, compare [68]. More generally, Kojadinovic and Marichal in [288] studied the moments and distributions of arbitrary Choquet discrete integrals.

Extended versions of functions from both of the above classes have been considered. In [436] the conditions on the triangle of coefficients choice for which a corresponding L-statistic has a limiting normal distribution is studied. For that we must assume certain weight generating schemes, compare Section 1.4.1. For instance, in [258] the convergence of weighted averages is studied, where there is one weight sequence $(c_1, c_2, \dots)$ and the statistic is of the form $\mathsf{F}(x_1, \dots, x_n) = \sum_{i=1}^{n} c_i x_i / \sum_{j=1}^{n} c_j$. On the other hand, like, e.g., in [68, 479], we may also assume that $\mathsf{F}(x_1, \dots, x_n) = \sum_{i=1}^{n} c_{i,n} x_i / \sum_{j=1}^{n} c_{j,n}$, where $c_{i,n} = C(i/n + 1)$ for some coefficient generating function $C :]0, 1[\to \mathbb{R}_{0+}$.

Discrete Sugeno integrals and other lattice polynomial functions. Marichal in [338] derived formulas for cumulative distribution functions and moments of lattice polynomial functions in the case of independent (but not necessarily identically distributed) random variables (real-valued ones). Note again that symmetric lattice polynomial functions are equivalent to sample quantiles. The case in which random variables are not necessarily independent was studied by Dukhovny in [173]. Also, Marichal and Kojadinovic studied the behavior of linear combinations of lattice polynomial functions in the case of uniformly distributed input data, see [342]. The i.i.d. case for weighted lattice polynomial functions was studied in [340]. Asymptotic behavior of the discrete Sugeno integral was studied by Gagolewski and Grzegorzewski in [213]. In particular, they showed the asymptotic normality of this fusion function and that it is a consistent estimator of some underlying probability distribution's characteristic of location.

Remark 4.20. Knowing the probabilistic behavior of fusion functions enables us to construct tools, e.g., aiming at statistical inference or decision making. For instance, a two-sample statistical hypothesis test for equality of Pareto distribution parameters based on the differences in the outputs of a particular Sugeno integral (the Hirsch index, see Section 5.4) was proposed by Gagolewski in [206].

Operations on random variables. Taking into account the above and other facts from probability theory, we may infer some new results concerning other classes of fusion functions. For instance, let φ be a strictly increasing and continuous function and assume that $Y = \varphi(X)$. Knowing that:

$$F_Y(u) = F_X(\varphi^{-1}(u)), \tag{4.12}$$

we may easily deduce the form of the cumulative distribution function of a quasi-arithmetic mean from the form of the c.d.f. of the arithmetic mean etc. What is more, note that under the current assumptions the density function is given by:

$$f_Y(u) = f_X(\varphi^{-1}(u))\frac{d}{du}\varphi^{-1}(u). \tag{4.13}$$

Basic arithmetic operations on independent random variables, see, e.g., [434], are given by:

$$f_{X+Y}(u) = (f_X \oplus f_Y)(u) = \int_{-\infty}^{+\infty} f_X(t) f_Y(u-t)\,dt, \tag{4.14}$$

$$f_{X-Y}(u) = (f_X \ominus f_Y)(u) = \int_{-\infty}^{+\infty} f_X(t) f_Y(t-u)\,dt, \tag{4.15}$$

$$f_{X\times Y}(u) = (f_X \otimes f_Y)(u) = \int_{-\infty}^{+\infty} \frac{f_X(t) f_Y(u/t)}{|t|}\,dt, \tag{4.16}$$

$$f_{X/Y}(u) = (f_X \oslash f_Y)(u) = \int_{-\infty}^{+\infty} \frac{f_X(t) f_Y(t/u)|t|}{u^2}\,dt. \tag{4.17}$$

Jaroszewicz and Korzeń in [260] study families of probability distributions closed under the above operations. Moreover, they develop a methodology for approximating arbitrary densities using piecewise Chebyshev interpolation. The PaCAL (probabilistic calculator) package for Python [295] is based on these results.

Orders in the space of random variables. There are many possible ways to introduce a partial order on the family of random variables, see, e.g., [311]. In particular, first order stochastic dominance is defined as:

$$X \preceq_{\text{st}} Y \text{ if and only if } (\forall x)\ F_X(x) \geq F_Y(x), \tag{4.18}$$

and the likelihood ratio order as:

$$X \preceq_{\text{lr}} Y \text{ if and only if } g(u) = \frac{f_Y(u)}{f_X(u)} \text{ is an increasing function of } u. \tag{4.19}$$

Linear orders may be introduced by considering, e.g., numerical characteristics of probability distributions such as the expected value (see Section 5.1). Moreover, it is not uncommon to consider various dissimilarity measures (which might not fulfill the triangle inequality), like the Kullback-Leibler divergence [303] and the Kolmogorov-Smirnov, Cramer-von Mises, or Anderson-Darling statistics which appear in the corresponding goodness-of-fit tests, see [435].

Randomness and fuzzy numbers. On a side note, we may also consider randomness and fuzziness together. In particular, Puri and Ralescu in [394] defined the concept of a random fuzzy variable as a mapping from a sample space Ω to the set of fuzzy numbers (see, e.g., [304] for one of the possible alternative approaches). In such a framework, e.g., Sinova and others [426–429] considered various types of median-like fusion functions for random intervals and random fuzzy numbers.

4.5 Aggregation of graphs and relations

Recall (compare Remark 1.176) that, at least for the purpose of this book, we may assume that there is a one-to-one correspondence between graphs and binary relations. Nevertheless, data fusion methods for the two classes of objects differ from each other as they most often serve much different practical purposes.

Aggregation of rankings and other relations. Suppose that $P = \{p_1, \ldots, p_k\}$ and let $\mathcal{L}(P)$ denote the set of all linear strict ordering relations on P. Our aim is to construct a fusion function $\mathsf{F} : \mathcal{L}(P)^n \to \mathcal{L}(P)$ that aggregates n linear ordering relations into one that is as much "concordant" with the inputs as possible. From now on we assume that the set of input orders $\mathbf{x} = (\sqsubset^{(1)}, \ldots, \sqsubset^{(n)})$ is fixed.

The construction of fair election methods continues to be of interest to many researchers since the 18th century. For instance, the famous *Borda count* assigns to each p_j, $j \in [k]$, a particular number of points relative to p_j's position in a ranking $\sqsubset^{(i)}$, $i \in [n]$, namely:

$$b_{i,j} = 1 + \left| \left\{ l \in [k] : p_l \sqsubset^{(i)} p_j \right\} \right|. \tag{4.20}$$

Then the position of p_j in the aggregated ranking is a function of the total number of points, $\bar{b}_j = \sum_{i=1}^{n} b_{i,j}$. The reader is referred to the extensive literature on the subject for a treatment of those kinds of data fusion methods at an appropriate level of detail, e.g., [15, 74, 123, 188, 321].

Nevertheless, we shall at least sketch two noteworthy classes of rank aggregation method construction. A *Kemeny-like optimal aggregation scheme*, compare [269], aims at finding a ranking $\sqsubset^* \in \mathcal{L}(P)$ which for some dissimilarity measure $\mathfrak{d}$ (e.g., a metric) has the property that:

$$\sum_{i=1}^{n} \mathfrak{d}(\sqsubset^*, \sqsubset^{(i)}) \le \sum_{i=1}^{n} \mathfrak{d}(\sqsubset, \sqsubset^{(i)}), \tag{4.21}$$

for all $\sqsubset \in \mathcal{L}(P)$. For instance, $\mathfrak{d}$ may be the already mentioned Dinu [156] rank distance or a function of the Kendall correlation coefficient τ [121], see also [152] for a different choice. This approach to rank fusion is in fact an instance of a penalty-based scheme. As most often exact algorithms for determining a Kemeny optimal solution are computationally intractable, various approximate methods are used in practice, e.g., ones that are based on evolutionary strategies (compare Algorithm 2.72).

A second approach is based on a notion of monotonicity. For instance, Rademaker and De Baets in [400] considered the following measure $\mathsf{S}_\mathbf{x} : \mathcal{L}(P)^2 \to [0 : n]$ of strength of support for a pair (p_i, p_j):

$$\mathsf{S}_\mathbf{x}(p_i, p_j) = \sum_{l=1}^{n} \mathbf{1}(p_i \sqsupset^{(l)} p_j), \tag{4.22}$$

see also [399]. For $p_i \neq p_j$ it holds that $\mathsf{S}_\mathbf{x}(p_i, p_j) + \mathsf{S}_\mathbf{x}(p_j, p_i) = n$. The authors suggest that the aggregated ranking $\sqsubset^*$ should fulfill the following monotonicity condition for all $p_i, p'_i, p_j, p'_j \in P$:

$$\begin{aligned}(p_i \sqsupseteq^* p'_i) \text{ and } (p_j \sqsupseteq^* p'_j) \text{ and } \big((p_i \sqsupset^* p'_i) \text{ or } (p_j \sqsupset^* p'_j)\big) \\ \implies \mathsf{S}_\mathbf{x}(p_i, p_j) \geq \mathsf{S}_\mathbf{x}(p'_i, p'_j).\end{aligned}$$

If the construction of such a ranking is not possible, it is allowed to "slightly modify" the values returned by $\mathsf{S}_\mathbf{x}$ so that a concordance becomes possible (note that the result might not be unique). Unfortunately, the procedure proposed in [400] requires that all the possible rankings in $\mathcal{L}(P)$ shall be considered. Nevertheless, the reader is already aware that, e.g., a genetic algorithm may quite easily be constructed to approximate the desired solution.

For an approach to aggregating arbitrary partial ordering relations, see, e.g., [398], in which pairwise preferences are learned through a majority-based voting process computed iteratively (using the notion of transitive closure) in such a way that cyclical and contradictory preferences are avoided.

Another interesting problem considers an aggregation of equivalence relations. For instance, Gionis, Mannila, and Tsaparas in [224] discuss the issue of *clustering aggregation*. Given n partitions $C_1, \ldots, C_n$ of the same data set X into an equal number of subsets, d, they review different methods to find a single d-partition that minimizes the total disagreement with the n input clusterings.

Aggregation of trees and other graphs. Graph representation of objects is particularly useful in various pattern recognition tasks, see, e.g., [60, 89]. We may be faced with a need to aggregate a set of graphs (possibly with labeled edges or nodes) when we need to determine a prototypical object in a set of similar glyphs in an optical character recognition task or to construct a k-means-like procedure for structurally described molecules. In such a case, we may rely on the notion of a penalty-based fusion function.

The following two classes of dissimilarity measures for graphs are most often referred to in the literature:

— metrics based on maximal common subgraphs or minimal common supergraphs, see, e.g., [90, 184, 459], for instance:

$$\mathfrak{d}(G_1, G_2) = 1 - \frac{|\mathrm{mcs}(G_1, G_2)|}{|G_1| \vee |G_2|}, \tag{4.23}$$

where $\mathrm{mcs}(G_1, G_2)$ denotes the maximal common subgraph of two input graphs and $|G|$ gives the number of vertices in a graph G,

— edit distance-based metrics, see, e.g., [60, 217, 440, 492], defined by considering the minimal number of node/edge relabeling, deletions, insertions (and possibly other types of edit operations) that are needed to transform a given graph to another one.

Interestingly, as shown by Bunke in [88], there exist graph-based and mcs-based distances which are equivalent to each other.

4.6 Aggregation in finite semimetric spaces

Let $X = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$ be a finite set and $(X, \mathfrak{d})$ denote a space equipped with a dissimilarity measure (a *semimetric*) $\mathfrak{d} : X \times X \to [0, \infty]$, i.e., one that fulfills:

— symmetry, i.e., $\mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \mathfrak{d}(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$ for all $i, j \in [n]$,

— $\mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{x}^{(i)}) = 0$ for all $i \in [n]$.

We would like to construct a fusion function F which aggregates all elements in X. Clearly, the output should be an element in X as well. Due to the high generality of the assumed model (which as a matter of fact is quite realistic), the set of possible operations that may be involved in the fusion process is limited: practically, we may only be looking for a penalty-based exemplar, see Section 2.5.2.

Let $\mathsf{D} : [0, \infty]^n \to [0, \infty]$ be a nondecreasing and idempotent fusion function such that $\mathsf{D}(n * 0) = 0$. We consider a fusion function like:

$$\mathsf{F}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = \operatorname*{arg\,min}_{\mathbf{y} \in X} \mathsf{D}\left(\mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{y}), \ldots, \mathfrak{d}(\mathbf{x}^{(n)}, \mathbf{y})\right). \tag{4.24}$$

If $\mathsf{D}(d_1, \ldots, d_n) = \sum_{i=1}^{n} d_i$, then we get a medoid, and if $\mathsf{D}(d_1, \ldots, d_n) = \bigvee_{i=1}^{n} d_i$, a seboid is obtained. Clearly, other choices of D are also possible and potentially useful.

A fast way to compute exemplars is crucial in, for instance, clustering large data sets. In practice, the costly part of all the algorithms to compute F involves the computation of $\mathfrak{d}$. This is the case of, e.g., long DNA sequences and the Levenshtein distance. Thus, our aim here is to discuss some possible approaches which keep the number of total calls to $\mathfrak{d}$ as small as possible.

Let us suppose that D is associative with neutral element e. The simplest approach to computing F is as follows.

Algorithm 4.21. *To compute* $\arg\min_{\mathbf{y}\in X} \mathsf{D}\left(\mathfrak{d}(\mathbf{x}^{(1)},\mathbf{y}),\ldots,\mathfrak{d}(\mathbf{x}^{(n)},\mathbf{y})\right)$ *in the case of associative* D *with neutral element* e*, proceed as follows:*

1. *Let* $\mathbf{d} = (n * e)$*;*
2. *For* $i = 1, 2, \ldots, n-1$ *do:*
   - 2.1. *For* $j = i+1, i+2, \ldots, n$ *do:*
     - 2.1.1. *Let* $d' = \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$*;*
     - 2.1.2. $d_i = \mathsf{D}(d_i, d')$*;*
     - 2.1.3. $d_j = \mathsf{D}(d_j, d')$*;*
3. *Return* $\mathbf{x}^{(i)}$ *as result, where* $i = \arg\min_{i\in[n]} d_i$*.*

It is easily seen that the above algorithm requires exactly $n(n-1)/2$ calls to $\mathfrak{d}$, thus, it does not adapt to input data at all. Therefore, we may consider the following algorithm.

Algorithm 4.22. *To compute* $\arg\min_{\mathbf{y}\in X} \mathsf{D}\left(\mathfrak{d}(\mathbf{x}^{(1)},\mathbf{y}),\ldots,\mathfrak{d}(\mathbf{x}^{(n)},\mathbf{y})\right)$ *in the case of associative* D *with neutral element* e*, proceed as follows:*

1. *Let* $b_d = \infty$*;*
2. *Let* $b_i = -1$*;*
3. *For* $i = 1, 2, \ldots, n$ *do:*
   - 3.1. $c_d = e$*;*
   - 3.2. *For* $j = 1, 2, \ldots, n$ *do:*
     - 3.1.1. *Let* $d = \mathfrak{d}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$*;*
     - 3.1.2. $c_d = \mathsf{D}(c_d, d)$*;*
     - 3.1.3. *If* $c_d \geq b_d$ *then break (go to step 3.3);*
   - 3.3. *If* $c_d < b_d$ *then:*
     - 3.3.1. $b_d = c_d$*;*
     - 3.3.2. $b_i = i$*;*
4. *Return* $\mathbf{x}^{(b_i)}$ *as result.*

This algorithm requires at least $2n$ but no more than n^2 calls to $\mathfrak{d}$. Its performance is thus strongly dependent on the type of input data, form of D, as well as the order of input elements. Note that it does not take into account the symmetry of $\mathfrak{d}$. Some savings would be possible at the cost of utilizing additional $O(n^2)$ memory, but for large n the use of such a cache is highly discouraged: if $n = 100{,}000$ and values of $\mathfrak{d}$ are stored as 8-byte `double` type, we would need

40 GB of RAM, which is way beyond memory limits of popular desktop PCs nowadays.

Example 4.23. Let us compare an average speedup (or slowdown) in terms of number of calls to $\mathfrak{d}$ of the second algorithm as compared to Algorithm 4.21. The averages are based on $M = 10$ Monte Carlo samples and in each considered scenario $n = 10{,}000$ input data items were aggregated.

metric $\mathfrak{d}$	**type of data in X**	**Alg. 4.22 speedup (medoid)**	**Alg. 4.22 speedup (seboid)**
Euclidean	normal distribution, $d = 200$	0.54	1796.1
Manhattan	normal distribution, $d = 200$	0.54	1519.6
Maximum	normal distribution, $d = 200$	0.55	1277.0
Dinu	ACTG sequences, $d = 200$	0.65	1681.8
Levenshtein	ispell Polish words dictionary	0.60	1646.9

It turns out that the use of the second procedure is particularly appealing when computing a seboid.

In certain applications (such as clustering), for large data sets we might be interested in a rough estimate of a set exemplar which is quite close to the true one but can be computed much faster. For that we propose the following procedure.

Algorithm 4.24. *To approximate* $\arg\min_{\mathbf{y}\in X} \mathsf{D}\left(\mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{y}), \ldots, \mathfrak{d}(\mathbf{x}^{(n)}, \mathbf{y})\right)$ *in the case of associative* D *with neutral element* e*, proceed as follows:*

1. *Let* c_i = *some random index in* $[n]$*;*
2. *Let* $\mathbf{v} = (n * 0)$*;*
3. *Let* $b_i = c_i$*;* *(current candidate)*
4. *Let* $b_d = \mathsf{D}\left(\mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{x}^{(c_i)}), \ldots, \mathfrak{d}(\mathbf{x}^{(n)}, \mathbf{x}^{(c_i)})\right)$*;*
5. $v_{c_i} = 1$*;* *(mark as visited)*
6. *Do:*
   - 6.1. *Let* $change = 0$*;*
   - 6.1. *For each* u_i *in (indices of* k*-nearest neighbors of* c_i*) do:*
     - 6.1.1. *If* $v_{u_i} = 1$ *then continue to step 6.1;*
     - 6.1.2. *Let* $u_d = \mathsf{D}\left(\mathfrak{d}(\mathbf{x}^{(1)}, \mathbf{x}^{(u_i)}), \ldots, \mathfrak{d}(\mathbf{x}^{(n)}, \mathbf{x}^{(u_i)})\right)$*;*
     - 6.1.3. $v_{u_i} = 1$*;* *(mark as visited)*
     - 6.1.4. *If* $u_d < b_d$ *then:* *(better candidate was found)*

6.1.4.1. $b_i = u_i$*;*
6.1.4.2. $b_d = u_d$*;*
6.1.4.3. $change = 1$*;*

6.2. $c_i = b_i$*;*

while $change = 1$*;*

7. *Return* $\mathbf{x}^{(b_i)}$ *as result.*

where k *is some fixed but small integer.*

This algorithm has been inspired by the steepest-descent optimization technique. Instead of computing the gradient (which in an arbitrary semimetric space is of course unavailable), an element's k nearest neighbors are taken into account. Starting from a randomly chosen point, we proceed in the direction which gives the best fit (inversely proportional to the value of D) until we find a local minimum. In order to increase the quality of the result, it is suggested to run the procedure a few times (note that the $\mathbf{v}$ vector should not be overwritten).

Figures A.21 and A.22 present a possible implementation of the algorithm which assumes by default $k = 5$ and 15 restarts. Numerical studies indicate that the procedure works reasonably well in the case of, e.g., $D(\mathbf{d}) = \sum_{i=1}^{n} d_i$, and $D(\mathbf{d}) = \sum_{i=1}^{n} d_i^2$, but is far from perfect in the case of a seboid search (which anyway can be performed very efficiently with Algorithm 4.23).

Example 4.25. Let us compute the speedup of an approximate medoid search in terms of the number of $\mathfrak{d}$ calls in Algorithm 4.22. Scenarios and experiment setup are identical to the those used in Example 4.23, we used $k = 5$ and 15 restarts.

metric $\mathfrak{d}$	**type of data in** X	**Alg. 4.24 speedup (medoid)**	**Alg. 4.24 rel. err. (medoid)**
Euclidean	normal distribution, $d = 200$	26.82	0.0001
Manhattan	normal distribution, $d = 200$	25.70	0.0003
Maximum	normal distribution, $d = 200$	25.47	0.0052
Dinu	ACTG sequences, $d = 200$	15.24	0.0065
Levenshtein	ispell Polish words dictionary	24.28	0.0134

We observe that the speedup is considerable while the relative error is kept small.

A different approximate algorithm – characterized by a competitive performance, but one which only works in the case of a medoid search task – is given by Micó and Oncina in [365] (note that our approach may also be used to improve the quality of its output). There is also an exact algorithm proposed by Juan

and Vidal in [264]. It may be applied in the cases when $\mathfrak{d}$ is a metric. Nevertheless, it does not perform well for high-dimensional data (due to the so-called *curse of dimensionality*, compare, e.g., [5]).

4.7 Aggregation of heterogeneous data

Aggregation of complex data sets consisting of heterogeneous variables (like those representing information coming from different types of sources and/or having incompatible representations) faces us with new challenges that perhaps were not present before. Nevertheless, it turns out that many of the methods we have already discussed are still valid in such a setting, compare [64, 447]. Other ones need to be adjusted accordingly or combined with other data fusion and data mining tools which start to serve their purpose when they are considered as a whole system.

Let us, however, note that the data fusion algorithms already presented are very general in their nature and thus can be described and examined via a plethora of formal methods and approaches. On the other hand, data science practitioners are aware of the fact that dealing with complex data sets sometimes may appear to be more art than science: each database often needs a customized treatment and it is not trivial to find frequently occurring, common patterns. Nevertheless, we shall at least try to explore limitations of the data fusion methods reviewed so far, suggest some heuristics to overcome them, as well as indicate few new types of data mining tasks, where their usage may be advantageous.

In the commencing sections of the second Chapter we discussed in detail data fusion methods to aggregate points in a d-dimensional space. We relied on an implicit assumption that the combined variables were homogeneous. In such a scenario, operations like rotations were fully justified. However, in a heterogeneous setting, this might not be the case. The easiest way to deal with data in complex domains is to apply simple componentwise fusion functions, that is, treat each of the variables independently. This is an imperfect solution, as any interactions between features cannot be taken into account in this way. Therefore, we may try to group variables of similar type and apply data fusion methods separately for each block. We can do the same with respect to clustered records that denote similar entities.

Penalty-based approaches may be quite powerful here too, especially if we have variables of mixed types, like categorical, ordinal, and numerical in one data set. Once a set of variables is partitioned, various dissimilarity measures may be introduced on each group, and then such measures may be aggregated (for instance, it is known that a conical combination of different metrics generates a new metric, etc.).

As usual, proper data wrangling – that is preprocessing, remapping, and reencoding – is a crucial initial stage of the data analysis process. An important step often consists of decorrelation of variables, e.g., via principal component analysis, correspondence analysis, or manifold learning procedures, see [242].

This can also lead to reduction in data dimensionality.

Fusion functions are also required in the process of improving data quality. For example, in the case of missing observations, it is customary to group (cluster) observations which represent similar entities and fill information that is not available with "averaged" results, compare [416].

Another area where they may be found useful deals with data deduplication and consolidation. That is, when removal of similar entities is needed. In such a case, we merge multiple redundant entries and replace them with aggregated ones, those that minimize information loss. For instance, Bronselaer, Szymczak, Zadrożny, and De Tré [85] develop a framework that during such a process takes into account a natural ordering relation that is learned dynamically from a data set. Moreover, in [84] a theoretical model for automated coreferent object detection and processing is proposed.

Let us also mention the record linkage task, see [59, 158, 472], which aim is to combine a set $\{D_1, \ldots, D_n\}$ of different, inconsistent, or non-unified databanks (e.g., SQL tables) into a single new database somehow. Typically, this is done by identifying all records in a database D_j that correspond to a record r in a databank D_i, $i \neq j$, either exactly (this may be done by simple join-like operations) or approximately in cases where data are contaminated by errors. In the latter setup, fast fuzzy matching algorithms are needed, including distance- and clustering-based ones, compare [447]. This typically involves the use of complex data structures such as vp- and kd-trees, GNAT, or similar [82, 316, 487], which speed up searching for similar objects.

Chapter 5

Numerical characteristics of objects

SYNTHETIC measures of diverse characteristics of objects are useful whenever there is a need to quantify how similar to or different from each other are given entities in terms of some carefully distinguished features. One such notion discussed already is a vector norm (see Definition 1.45). We shall see that in order to capture exactly a type of behavior or property that is of interest to a practitioner in a particular setting, we should rely on its proper mathematical axiomatization. In this chapter we are interested in exploring various ways to numerically characterize probability distributions, spread of numerical sequences (in their entirety), the degree of decision makers' consensus, economic inequality or poverty, entropy, empirical distribution shape, fuzzy numbers, as well as fusion functions themselves. We end the discussion with the notion of a checksum function, which shall appear different in its very nature from all the other measures.

5.1 Characteristics of probability distributions

The development of currently widely used measures of data central tendency and variability is inevitably connected with the history of probability and statistics. Perhaps the first official (published) use of the term *standard deviation* (in the context of probability) is due to Pearson [385, Part I]:

> *Let the equation to the probability-curve be* $y' = \frac{1}{\sigma\sqrt{2\pi}} e^{-x^2/(2\sigma^2)}$. *Then* σ *will be termed its* standard-deviation *(error of mean square). [385, page 80]*

However, it is known that the terms "root mean squared error" and "mean error" were already used by Gauss.

On the other hand, the term *variance* was probably first defined in the paper by R.A. Fisher [191]:

> *It is therefore desirable in analyzing the causes of variability to deal with the square of the standard deviation as the measure of variability. We shall term this quantity the* Variance *(...). [191, page 399]*

Please note that both quotations discuss in fact the underlying probability distribution characteristic, and not the (observed) sample-based estimates. The debate leading to the acceptance of a proper distinction between the objects being of interest of probability theory, on the one hand, and statistics, on the other, engaged many leading researchers for many years in the first decades of the 20th century[1]. The need to discriminate between a (probabilistic) population with its characteristics (such as expected value μ or variance σ^2), and a (statistical, observed) sample from which we may calculate the characteristics' estimates (like mean $\bar{\mathbf{x}}$ or *sample* variance s^2) is explained, e.g., in Fisher's paper [192]:

> *(...) it has happened that in statistics a purely verbal confusion has hindered the distinct formulation of statistical problems; for it is customary to apply the same name,* mean, standard deviation, correlation coefficient*, etc., both to the true value which we should like to know, but can only estimate, and to the particular value at which we happen to arrive by our methods of estimation; so also in applying the term probable error, writers sometimes would appear to suggest that the former quantity, and not merely the latter, is subject to error.*

Moreover, Fisher in the same paper [192] considered two estimates of the (population's) standard deviation σ (in some statistical model), namely the *mean error*:

$$\mathsf{ME}(\mathbf{x}) = \frac{1}{n}\sqrt{\frac{\pi}{2}}\sum_{i=1}^{n}|x_i - \mathsf{AMean}(\mathbf{x})|, \tag{5.1}$$

and the *mean squared error* defined as:

$$\mathsf{MSE}(\mathbf{x}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mathsf{AMean}(\mathbf{x}))^2}. \tag{5.2}$$

This paper is a beautiful early example of a comparative study concerning the usage of two different sample statistics that shall measure the same quantity.

In this section our main focus is on various methods that can be used to measure central tendency or dispersion of probability distributions. Note that in Section 4.4 we considered fusion functions that act on *random data* and this

[1] According to the on-line encyclopedia "Earliest Known Uses of Some of the Words of Mathematics" (maintained by J. Aldrich, see http://jeff560.tripod.com/mathword.html, see also "The Oxford Dictionary of Statistical Terms"): *Although Student (1908) had used the phrases, "mean of the population" and "mean of the sample", it was not until the 1930s that such terms as sample mean or population standard deviation became prominent.*

time we are interested in functions that are used to numerically summarize some aspects of the underlying probability distribution's behavior.

Then, we shall consider the link between the two approaches. More precisely, we concentrate on the properties that a statistic (a fusion function acting on random data) should fulfill in order to call it an *estimator* of a probability distribution characteristic.

5.1.1 Measures of location

Denote by $\mathcal{D}_d$ the set of probability distributions in $\mathbb{R}^d$, $d \geq 1$. Note that if a random variable X is f-distributed, that is $X \sim f \in \mathcal{D}_d$, then we shall also denote this fact by $X \in \mathcal{D}_d$ for brevity. Moreover, if $\varphi : \mathcal{D}_d \to Z$ for some set Z, then instead of writing $\varphi(f)$ we shall also use the notation $\varphi(X)$.

Oja in [378] considered the following axiomatization of measures of location, which is a multivariate generalization of a model introduced by Bickel and Lehmann in [56].

Definition 5.1. We call $\mathsf{L} : \mathcal{D}_d \to \mathbb{R}^d$ a measure of location in the Oja sense, whenever:

(a) for any $X, Y \in \mathcal{D}_d$ if $X \preceq_{\text{st}} Y$, then $\mathsf{L}(X) \leq \mathsf{L}(Y)$,

(b) for all matrices $\mathbf{A} \in \mathbb{R}^{d \times d}$ of full rank, all $\mathbf{t} \in \mathbb{R}^d$, and $X \in \mathcal{D}_d$ such that $\mathbf{A}X + \mathbf{t} \in \mathcal{D}_d$ it holds that $\mathsf{L}(\mathbf{A}X + \mathbf{t}) = \mathbf{A}\mathsf{L}(X) + \mathbf{t}$.

In other words, a measure of location is first order stochastic dominance-monotone and affine equivariant.

Note that if the distribution of X is symmetric about $\boldsymbol{\mu}$, that is $\boldsymbol{\mu} - X$ has the same distribution as $X - \boldsymbol{\mu}$, then $\mathsf{L}(X) = \boldsymbol{\mu}$. In other words, if $\mathcal{D}_d$ is a class consisting solely of symmetrical probability distributions, then all measures of location coincide.

Apart from the expected value, $\mathbb{E}\, X$, also, e.g., the population version of the Oja median is an example of a location measure.

5.1.2 Measures of dispersion

Bickel and Lehmann in [57] considered measures of dispersion for a family of symmetric univariate probability distributions $\mathcal{D}_1$. We consider X less dispersed than Y, denoted $X \preceq_{\text{d}} Y$, whenever $|X - \mu_X| \preceq_{\text{st}} |Y - \mu_Y|$, where μ_X and μ_Y denote the points of symmetry of X and Y, respectively. Then $\mathsf{S} : \mathcal{D}_1 \to [0, \infty]$ is called a dispersion (scatter) measure, whenever:

(a) for all $X, Y \in \mathcal{D}_1$, if $X \preceq_{\text{d}} Y$, then $\mathsf{S}(X) \leq \mathsf{S}(Y)$,

(b) for all $s, t \in \mathbb{R}$, $X \in \mathcal{D}_1$, if $sX + t \in \mathcal{D}_1$, then $\mathsf{S}(sX + t) = |s|\mathsf{S}(X)$.

Equivalently, dispersion measures are $\preceq_{\mathrm{d}}$-monotone, scale equivariant, and translation invariant.

The presented notion has been generalized by Oja in [378]. He defined $\mathsf{S} : \mathcal{D}_d \to [0, \infty]$ to be a scatter measure, if it fulfills generalized $\preceq_{\mathrm{d}}$-monotonicity (the concept is based on areas of appropriate multidimensional simplices), as well as such that $\mathsf{S}(\mathbf{A}X + \mathbf{t}) = |\det(\mathbf{A})|\mathsf{S}(X)$. In particular, this class includes the following measures for any $p > 0$:

— generalizations of the unidimensional standard deviation:

$$\operatorname{Sd} X = \sqrt{\operatorname{Var} X} = \sqrt{\mathbb{E}\, X^2 - (\mathbb{E}\, X)^2} = \sqrt{(\mathbb{E}(\mathbb{E}\, X - X)^2)},$$

such as:

$$\mathsf{S}(X) = \sqrt[p]{\mathbb{E}\big(\operatorname{vol}(\operatorname{CH}(\mathbb{E}X, X_1, \ldots, X_d))\big)^p},$$

— generalizations of the Gini mean difference such as:

$$\mathsf{S}(X) = \sqrt[p]{\mathbb{E}\big(\operatorname{vol}(\operatorname{CH}(X_1, \ldots, X_{d+1}))\big)^p},$$

where $f \in \mathcal{D}_d$ and $X, X_1, \ldots, X_{d+1}$ i.i.d. f.

Additionally, Bickel and Lehmann in [58] considered measures of spread for univariate but not necessarily symmetric distributions.

5.1.3 Point estimation

Suppose that S is a statistic and that F_ϑ is a probability distribution characterized by some parameter ϑ. Assume that $\mathbf{X} = (X_1, \ldots, X_n)$ is a sequence of random variables following F_ϑ (most often they are considered to be independent). The aim of point estimation, see, e.g., [310, 423], is to determine whether $\mathsf{S}(X_1, \ldots, X_n)$ may be used somehow to *guess* – in an educated manner – the value of ϑ. As ϑ is a fixed value and $\mathsf{S}(X_1, \ldots, X_n)$ is a random variable, there are many possible ways to relate these two objects. In particular, we may be interested in measuring an estimator's:

— *Bias* or expected systematic error, i.e.:

$$\mathsf{Bias}_\vartheta(\mathsf{S}) = \mathbb{E}\left(\mathsf{S}(\mathbf{X}) - \vartheta\right). \tag{5.3}$$

Note that if $\mathsf{Bias}_\vartheta(\mathsf{S}) = 0$, we call S an *unbiased estimator* of ϑ.

— *Mean squared error*, i.e.:

$$\mathsf{MSE}_\vartheta(\mathsf{S}) = \mathbb{E}\left(\mathsf{S}(\mathbf{X}) - \vartheta\right)^2. \tag{5.4}$$

It is well-known that $\mathsf{MSE}_\vartheta(\mathsf{S}) = \operatorname{Var}\mathsf{S}(\mathbf{X}) + (\mathsf{Bias}_\vartheta(\mathsf{S}))^2$.

— *Efficiency*, which is equal to 1 whenever S is unbiased and has the smallest possible mean squared error (and hence variance) among all unbiased estimators of ϑ.

For instance, it may be shown that if $(X_1, \ldots, X_n)$ is a sample of i.i.d. random variables with expectation of μ and variance of σ^2, then the sample variance given by:

$$\mathsf{Var}(\mathbf{X}) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \mathsf{AMean}(\mathbf{X}))^2 \tag{5.5}$$

is an unbiased estimator of σ^2. Moreover, the arithmetic mean is an unbiased estimator of μ. If, additionally, the random variables are normally distributed, then AMean is of efficiency 1. In such a case, the sample median is an unbiased estimator of μ too, but yet not as efficient.

Additionally, asymptotic properties (for arbitrarily large n) may also be studied. If $(X_1, X_2, \ldots)$ is a sequence of F_ϑ-distributed random variables, these include:

— asymptotic unbiasedness,

— asymptotic efficiency,

— asymptotic normality,

— consistency, which holds if $\lim_{n\to\infty} \mathrm{P}_\vartheta(|\mathsf{S}(X_1, \ldots, X_n) - \vartheta| < \varepsilon) = 1$ for all $\varepsilon > 0$,

and so on. For instance, the sample standard deviation, SD, is only an asymptotically unbiased estimator of the population standard deviation, σ.

From the described perspective, it is not unusual to take a unidimensional fusion function F, treat it as a statistic, and answer questions such as:

— What does F estimate?

— How well does it perform in doing so?

in particular probability models. Such an approach may provide a new insight into the existing fusion functions (compare Section 4.4 too).

5.2 Spread measures

Many introductory textbooks on applied statistics and academic lectures on the subject include a review of the so-called descriptive statistics, i.e., methods for summarizing quantitative unidimensional data for performing exploratory data analysis. Most often such methods are divided into at least two classes (see [3, Chapter 1] and, e.g., [132]):

1. *Measures of central tendency* (also known as measures of location or centrality of observations); e.g., sample quantiles (including median, min, and max), arithmetic mean, mode, trimmed and Winsorized mean etc.

2. *Measures of variability* (or data spread), e.g., range, interquartile range, variance, standard deviation.

As we noted in the first chapter, aggregation theory classically focuses on (among others) the broadly-conceived means. However, we often need a very different kind of a proper synthesis of multidimensional numeric data into a single number – the one that falls into the second category above.

It turns out that popular measures of data variability may be divided further into the following subclasses:

2a. *Measures of absolute data spread*, e.g., standard deviation, interquartile range, median absolute deviation. In this case, an absolute spread measure V may accompany an aggregation function A in order to state that a numeric list $\mathbf{x}$ is concisely described as $\mathsf{A}(\mathbf{x}) \pm \mathsf{V}(\mathbf{x})$.

2b. *Measures of relative data spread* (e.g., Gini coefficient, coefficient of variation), which are dependent on the order of magnitude of a numeric list's elements. For instance, imagine that we have two groups of people. The first group consists of $(1, 2, 3)$-year-olds and the second one of $(101, 102, 103)$-year-olds. Intuitively, the relative spread of age in the first group is greater than that of the second group.

In this section we would like to focus on measures of absolute data spread from the perspective of aggregation theory. For that, we shall properly axiomatize this class so that we establish exactly our universe of discourse.

Remark 5.2. Pitman in [392] claimed that the function C, used to estimate the scale parameter c in his simple translate-scale model (see Remark 1.56), should satisfy the conditions:

(c1) $\mathsf{C}(x_1, \ldots, x_n) \geq 0$, (nonnegativity)

(c2) $\mathsf{C}\left(\frac{x_1+\lambda}{\mu}, \ldots, \frac{x_n+\lambda}{\mu}\right) = \frac{\mathsf{C}(x_1,\ldots,x_n)}{\mu}$, for all $\lambda \in [-\infty, \infty]$ and $\mu > 0$. (scale equivariance and translation invariance)

Unfortunately, his setting does not serve our purposes: it is too weak. Let:

$$\mathsf{S}(\mathbf{x}) = \begin{cases} 0 & \text{for } \mathbf{x} = (n * c) \text{ for some } c, \\ \dfrac{\sum_{i=1}^{n}(x_i - \bar{\mathbf{x}})^2}{\sum_{i=1}^{n}(x_i - x_{(1)})} & \text{otherwise.} \end{cases}$$

It is easily seen that S is nonnegative, translation invariant, and $\mathsf{S}(s\mathbf{x}) = s\mathsf{S}(\mathbf{x})$ for all $s \geq 1$. However, it holds that $\mathsf{S}(0, 4, 100) \simeq 61.64 > \mathsf{S}(0, 10, 107) \simeq 59.71$, which is counter-intuitive. Moreover, it may be easily seen that some "classical" spread measures, e.g., the sample variance, do not fulfill (c2).

5.2.1 Measures of absolute spread for unidimensional data

Here we shall rather rely on the axiomatization of measures of absolute data spread which was proposed by Gagolewski in [209].

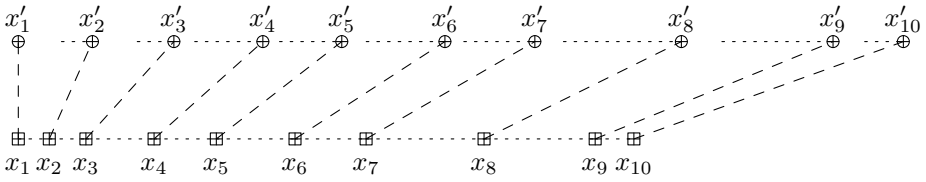


Figure 5.1. Two exemplary numeric lists with different spreads: $\mathbf{x} \preccurlyeq_n \mathbf{x}'$.

Definition 5.3. For some $\mathbb{I} = [a, b]$, given $\mathbf{x}, \mathbf{x}' \in \mathbb{I}^n$, we write $\mathbf{x} \preccurlyeq_n \mathbf{x}'$ and say that $\mathbf{x}$ *has not greater absolute spread than* $\mathbf{x}'$, if and only if for all $i, j \in [n]$ it holds:

$$(x_i - x_j)(x'_i - x'_j) \geq 0 \text{ and } |x_i - x_j| \leq |x'_i - x'_j|. \tag{5.6}$$

Please note that $\preccurlyeq_n$ is a preorder on $\mathbb{I}^n$, that is, a relation that is reflexive and transitive. What is more, it is not necessarily total, i.e., not all vectors are comparable with each other.

Additionally, whether $\preccurlyeq_n$ holds for given $\mathbf{x}, \mathbf{x}'$ depends on how the elements in both vectors are jointly ordered. The left side of (5.6) implies that if $\mathbf{x} \preccurlyeq_n \mathbf{x}'$, then $\mathbf{x}, \mathbf{x}'$ are comonotonic.

Figure 5.1 illustrates two vectors: $\mathbf{x}$ and its modified version $\mathbf{x}'$ with increased distances between consecutive elements.

Remark 5.4. Let us study how $\preccurlyeq_n$ behaves under scaling and translation of elements in a given vector.

It is easily seen that for all $s \geq 1$ and $\mathbf{x} \in \mathbb{I}^n$ such that $s\mathbf{x} \in \mathbb{I}^n$ we have $\mathbf{x} \preccurlyeq_n s\mathbf{x}$. Additionally, for all $t \in \mathbb{R}$ for which $t + \mathbf{x} \in \mathbb{I}^n$ it holds $\mathbf{x} \preccurlyeq_n t + \mathbf{x}$ and, at the same time, $t + \mathbf{x} \preccurlyeq_n \mathbf{x}$. Thus, $\preccurlyeq_n$ is not antisymmetric.

What is more, for all $c \in \mathbb{I}$, $(n * c)$ is a minimal element of $(\mathbb{I}^n, \preccurlyeq_n)$, i.e., for any $\mathbf{x}$ we have $(n * c) \preccurlyeq_n \mathbf{x}$. This relation is also convex: for all $\mathbf{x}, \mathbf{x}', \alpha \in [0, 1]$ it holds $\mathbf{x} \preccurlyeq_n \alpha\mathbf{x} + (1 - \alpha)\mathbf{x}' \preccurlyeq_n \mathbf{x}'$ whenever $\mathbf{x} \preccurlyeq_n \mathbf{x}'$.

Let us proceed with the definition of objects in which we have a special interest in this section.

Definition 5.5 ([209]). A *spread measure* is a mapping $\mathsf{V} : \mathbb{I}^n \to [0, \infty]$ such that:

(v1) for each $\mathbf{x} \preccurlyeq_n \mathbf{x}'$ it holds $\mathsf{V}(\mathbf{x}) \leq \mathsf{V}(\mathbf{x}')$,

(v2) for any $c \in \mathbb{I}$ it holds $\mathsf{V}(n * c) = 0$.

Note that the first characteristic property implies that each spread measure is translation invariant. Moreover, for all $s \geq 1$ and $\mathbf{x} \in \mathbb{I}^n$ such that $s\mathbf{x} \in \mathbb{I}^n$ it holds $\mathsf{V}(\mathbf{x}) \leq \mathsf{V}(s\mathbf{x})$.

In [209] it has been shown that this class includes, among others, the following spread measures:

— $\mathsf{Var}(\mathbf{x}) = \frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \mathsf{AMean}(\mathbf{x})\right)^2$, (*sample variance*)

— $\mathsf{SD}(\mathbf{x}) = \sqrt{\mathsf{Var}(\mathbf{x})}$, (*standard deviation*)

— $\mathsf{Range}(\mathbf{x}) = \mathsf{Max}(\mathbf{x}) - \mathsf{Min}(\mathbf{x})$, (*range*)

— $\mathsf{IQR}(\mathbf{x}) = \mathsf{Q}_{0.75}(\mathbf{x}) - \mathsf{Q}_{0.25}(\mathbf{x})$, (*interquartile range*)

— $\mathsf{MAD}(\mathbf{x}) = 1.4826\mathsf{Median}(|\mathbf{x} - \mathsf{Median}(\mathbf{x})|)$, (*median absolute deviation*)

— $\mathsf{ME}(\mathbf{x}) = \frac{1}{n}\sqrt{\frac{\pi}{2}}\sum_{i=1}^{n}|x_i - \mathsf{AMean}(\mathbf{x})|$, (*Fisher's mean error*)

that is functions widely used in exploratory data analysis (all of them are symmetric). Note that the sample variance, standard deviation, mean error, and range are 3-incremental fusion functions.

Proposition 5.6. *Let* V *be a spread measure such that* $\sup_{\mathbf{x}\in\mathbb{I}^n}\mathsf{V}(\mathbf{x}) = u$. *Then for each nondecreasing function* $\varphi : [0,u] \to [0,(b-a)]$ *such that* $\varphi(0) = 0$, $\varphi \circ \mathsf{V}$ *is a spread measure too.*

Taking the above into account, the following further classes of fusion functions (together with their monotone transforms) may be distinguished:

— $\mathsf{V}(\mathbf{x}) = \sum_{i=1}^{n}\sum_{k=1}^{n}|x_i - x_k|^p$ for some $p \geq 1$, in particular, the sample variance:

$$\mathsf{Var}(\mathbf{x}) = \frac{1}{2n(n-1)}\sum_{i=1}^{n}\sum_{k=1}^{n}\left(x_i - x_k\right)^2 \tag{5.7}$$

and the *Gini mean difference*:

$$\mathsf{MD}(\mathbf{x}) = \frac{1}{n\,(n-1)}\sum_{i=1}^{n}\sum_{k=1}^{n}|x_i - x_k|, \tag{5.8}$$

— $\mathsf{V}(\mathbf{x}) = \mathsf{A}\left(|x_1 - \mathsf{Q}_\alpha(\mathbf{x})|, \ldots, |x_n - \mathsf{Q}_\alpha(\mathbf{x})|\right)$ for some n-ary classical aggregation function A and $\alpha \in [0,1]$, e.g., MAD, IQR, and Range,

— $\mathrm{WD}_2\mathrm{WAM}$ spread measures of the form:

$$\mathsf{V}(\mathbf{x}) = \sum_{i=1}^{n} w_i \left(x_i - \sum_{j=1}^{n} w_j x_j\right)^2 \tag{5.9}$$

for some weighting vector $\mathbf{w}$ as well as their symmetrized counterparts ($\mathrm{WD}_2\mathrm{OWA}$ operators), e.g., the sample variance,

— $\mathrm{WD}_1\mathrm{WAM}$ spread measures $\mathsf{V}(\mathbf{x}) = \sum_{i=1}^n w_i \left| x_i - \sum_{j=1}^n w_i x_j \right|$ and the corresponding $\mathrm{WD}_1\mathrm{OWA}$ operators, e.g., the Fisher mean error,

— $\mathrm{WD}_\infty\mathrm{WAM}$ operators $\mathsf{V}(\mathbf{x}) = \max_{i=1,\ldots,n} \left| x_i - \sum_{j=1}^n w_j x_j \right|$ as well as the corresponding $\mathrm{WD}_\infty\mathrm{OWA}$ operators.

Note that in [207] Gagolewski considered normalized versions of some of the above spread measures classes which can be used in decision making. They attain the greatest possible value equal to $(b-a)$ (in particular, 1 if $\mathbb{I} = [0,1]$). For example:

$$\mathsf{NWD}_2\mathsf{WAM}_{\mathbf{w}}(\mathbf{x}) = \frac{\sqrt{\sum_{i=1}^n w_i \left(x_i - \sum_{j=1}^n w_j x_j \right)^2}}{(b-a)\sqrt{p(1-p)}}, \tag{5.10}$$

where $p = \max_{A\subseteq[n], \sum_{i\in A} w_i \leq 0.5} \sum_{i\in A} w_i$, and:

$$\mathsf{NWD}_1\mathsf{WAM}_{\mathbf{w}}(\mathbf{x}) = \frac{\sum_{i=1}^n w_i \left| x_i - \sum_{j=1}^n w_j x_j \right|}{2p(1-p)(b-a)}, \tag{5.11}$$

where $p = \max_{A\subseteq[n], \sum_{i\in A} w_i \leq 0.5} \sum_{i\in A} w_i$.

Let us proceed with an appealing characterization of measures of absolute spread. For any given $\mathbf{x} \in \mathbb{I}^n$, let $\mathsf{diff}(\mathbf{x}) = (x_{(2)} - x_{(1)}, \ldots, x_{(n)} - x_{(n-1)}) \in [0, (b-a)]^{n-1}$ denote the *iterated difference* between consecutive ordered components of a given vector. Please note that such a function is available in some programming languages: in particular, it may be computed by calling `diff(sort(x))` in R. We see that if $\boldsymbol{\delta} = \mathsf{diff}(\mathbf{x})$, then $0 \leq \delta_i \leq b-a$ and $\sum_{i=1}^{n-1} \delta_i \leq b-a$. Intuitively, if $\mathbf{x}$ is already ordered, then this operation may be viewed as a kind of "vector differentiation". On the other hand, for $\tilde{\mathbf{x}} = \mathsf{cumsum}(x_{(1)}, \boldsymbol{\delta}) = (x_{(1)}, x_{(1)} + \delta_1, x_{(1)} + \delta_1 + \delta_2, \ldots, x_{(1)} + \delta_1 + \cdots + \delta_n)$ denoting the *cumulative sum* of $\hat{\boldsymbol{\delta}} = (x_{(1)}, \boldsymbol{\delta})$ we have $x_{(i)} = \tilde{x}_i$, $x_i = \tilde{x}_{\sigma^{-1}(i)}$, where σ is such that $\mathbf{x} \in \mathbb{I}^n_\sigma$. Thus, $\mathbf{x}$ may be reconstructed from $x_{(1)}, \boldsymbol{\delta}$, and σ.

We are now in a position to provide an equivalent definition of the relation defined by Equation (5.6).

Lemma 5.7 ([209]). *For any $\mathbf{x}, \mathbf{x}' \in \mathbb{I}^n$ it holds $\mathbf{x} \preccurlyeq_n \mathbf{x}'$ if and only if $\mathbf{x}, \mathbf{x}'$ are comonotonic and $\mathsf{diff}(\mathbf{x}) \leq_{n-1} \mathsf{diff}(\mathbf{x}')$.*

Therefore, we have what follows.

Theorem 5.8 ([209]). *$\mathsf{V} : \mathbb{I}^n \to [0, \infty]$ is a spread measure if and only if the following conditions are valid:*

(v1') *for each comonotonic $\mathbf{x}, \mathbf{x}'$ such that $\mathsf{diff}(\mathbf{x}) \leq_{n-1} \mathsf{diff}(\mathbf{x}')$ we have $\mathsf{V}(\mathbf{x}) \leq \mathsf{V}(\mathbf{x}')$,*

(v2') $\inf_{\mathbf{x}\in\mathbb{I}^n} \mathsf{V}(\mathbf{x}) = 0$.

Corollary 5.9. *For any* $\mathsf{V}: \mathbb{I}^n \to [0,\infty]$, $\mathsf{V}|_\sigma$ *fulfills (v1) and (v2) if and only if there exists* $\tilde{\mathsf{A}}: [0, b-a]^{n-1} \to [0,\infty]$ *such that* $\mathsf{V}|_\sigma(\mathbf{x}) = \tilde{\mathsf{A}}(\mathsf{diff}(\mathbf{x}))$ *is nondecreasing and lower endpoint-preserving.*

We see that symmetric absolute spread measures are nothing more than aggregation functions computed on iterated differences of an input vector.

5.2.2 Measures of relative spread

As indicated in [209], some "normalized" measures of *relative* spread may also be considered. At the most general level, these are functions of the form:

$$\mathsf{S}(\mathbf{x}) = \frac{\mathsf{V}(\mathbf{x})}{\mathsf{A}(\mathbf{x})}, \tag{5.12}$$

where V is an absolute spread measure, and A is an aggregation function.

For instance, the well known (unit-free) *Gini coefficient*, defined as:

$$\mathsf{Gini}(\mathbf{x}) = \frac{\mathsf{MD}(\mathbf{x})}{2\mathsf{AMean}(\mathbf{x})} \tag{5.13}$$

is definitely not a measure of absolute spread. This is because it is not even translation invariant: we have $\mathsf{Gini}(0,2,4) = 2/3$, and $\mathsf{Gini}(2,4,6) = 1/3$. Moreover, even though $(0,2,4) \preccurlyeq_n (0,3,5)$, we have $\mathsf{G}(0,3,5) = 5/8 < 2/3 = \mathsf{G}(0,2,4)$. A similar observation may be made about the so-called *coefficient of variation*:

$$\mathsf{CV}(\mathbf{x}) = \frac{\mathsf{SD}(\mathbf{x})}{\mathsf{AMean}(\mathbf{x})}. \tag{5.14}$$

Both functions take into account the order of magnitude of the observations, and are ratio scale invariant (i.e., $\mathsf{S}(s\mathbf{x}) = \mathsf{S}(\mathbf{x})$ for all $s > 0$) as well as continuous but not translation invariant.

5.2.3 Spread measures for multidimensional data

Similarly as in Chapter 2, let us again assume that we are given $\mathbf{X} \in (\mathbb{R}^d)^n$. As noted, e.g., in [324], there are two ways to quantify dispersion of a multivariate data set: as a matrix or as a scalar. The latter is of course much easier to construct and fits the overall setting established in this chapter. Nevertheless, let us at least mention that, e.g., the sample *covariance matrix*, given by:

$$\mathsf{Cov}(\mathbf{X}) = \frac{1}{n-1} \left(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X})\right) \left(\mathbf{X} - \mathsf{CwAMean}(\mathbf{X})\right)^T \in \mathbb{R}^{d\times d},$$

can reveal other useful information on a dataset, such as the orientation of the empirical probability mass distribution and the dispersion of individual variates or covariates.

In a recent contribution, Kołacz and Grzegorzewski [289] considered multidimensional spread measures defined as functions $\mathsf{V}: (\mathbb{R}^d)^n \to [0,\infty]$ that are:

— symmetric,

— translation and rotation invariant,

— homogeneous, i.e., there exists a nondecreasing function $\varphi : [0, \infty[\to [0, \infty[$ such that $\mathsf{V}(s\mathbf{X}) = \varphi(s)\mathsf{V}(\mathbf{X})$ for all $s > 0$ and $\mathbf{X} \in (\mathbb{R}^d)^n$.

— such that $\mathsf{V}(n * \mathbf{x}) = 0$ for all $\mathbf{x} \in \mathbb{R}^d$.

Similarly as the Pitman [392] axiomatization of a scale parameter estimate, their setting seems to be too mild, as it only concerns uniform scaling in each direction, translation, and rotation transforms. It would be informative, if it referred to some ordering relation. Nevertheless, this axiomatization is a good starting point for future research on the topic – to our best knowledge there are no alternatives to this proposal in the literature yet. Interestingly, the authors explore the relationships between spread measures for vectors of different arities and functions that are generated via particular so-called multidistances [346, 347].

Remark 5.10. The Oja simplex volume-approach (see [378] and Section 5.1.2) gives one possibility for generalizing the $\preccurlyeq_n$ relation defined in [209]. Note that for $d = 1$ the condition $|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}| \leq |\mathbf{x}'^{(i)} - \mathbf{x}'^{(j)}|$ presented in Equation (5.6) may be written also as:

$$\operatorname{vol}\left(\operatorname{CH}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})\right) \leq \operatorname{vol}\left(\operatorname{CH}(\mathbf{x}'^{(i)}, \mathbf{x}'^{(j)})\right),$$

which now can be generalized for any d as:

$$\operatorname{vol}\left(\operatorname{CH}(\mathbf{x}^{(i_1)}, \ldots, \mathbf{x}^{(i_{d+1})})\right) \leq \operatorname{vol}\left(\operatorname{CH}(\mathbf{x}'^{(i_1)}, \ldots, \mathbf{x}'^{(i_{d+1})})\right),$$

where $i_1, \ldots, i_{d+1} \in [n]$. Regardless of some problems with defining comonotonicity (which can be quite easily bypassed), a fusion function monotone with respect to the above partial ordering is automatically translation and rotation invariant. Moreover, uniform scaling in each direction shall never lead to a decrease in its output.

Here are a few particular classes and/or general construction methods for dispersion measures.

— Unidimensional spread measures may be generalized to any d via projection pursuit, see [253]. This is because we may apply all possible one-dimensional projections of the data set and compute the univariate V. For instance:

$$\mathsf{V}'(\mathbf{X}) = \sup_{\|\mathbf{u}\|=1} \mathsf{V}(\mathbf{u}^T\mathbf{X}),$$

which gives the maximal possible directional variance (compare the Principal Component Analysis method), or:

$$\mathsf{V}''(\mathbf{X}) = \int_{\|\mathbf{u}\|=1} \mathsf{V}(\mathbf{u}^T\mathbf{X})\, d\mathbf{u},$$

which gives the averaged dispersion, see Figure 5.2 for a graphical illustration.

— Given a semimetric $\mathfrak{d}$ on $\mathbb{R}^d$ and a fusion function $\mathsf{F} : [0,\infty]^{n(n-1)/2} \to [0,\infty]$, compute:

$$\mathsf{V}'(\mathbf{X}) = \mathsf{F}\left(\mathfrak{d}(\mathbf{x}^{(1)},\mathbf{x}^{(2)}), \mathfrak{d}(\mathbf{x}^{(1)},\mathbf{x}^{(3)}), \ldots, \mathfrak{d}(\mathbf{x}^{(n-1)},\mathbf{x}^{(n)})\right),$$

in particular F can be the arithmetic mean.

— Given a semimetric $\mathfrak{d}$ on $\mathbb{R}^d$, a fusion function $\mathsf{A} : (\mathbb{R}^d)^n \to \mathbb{R}^d$, and a fusion function $\mathsf{F} : [0,\infty]^n \to [0,\infty]$, compute:

$$\mathsf{V}'(\mathbf{X}) = \mathsf{F}\left(\mathfrak{d}(\mathbf{x}^{(1)},\mathsf{A}(\mathbf{X})), \ldots, \mathfrak{d}(\mathbf{x}^{(n-1)},\mathsf{A}(\mathbf{X}))\right),$$

in particular F can be the quadratic mean or the Max function and A – the componentwise arithmetic mean (centroid).

Moreover, Liu, Parelius, and Singh in [324] consider a few data depth-based dispersion measures.

Remark 5.11. Various methods for measuring dispersion of directional data exist as well. Given a circular data sample $\vartheta_1, \ldots, \vartheta_n \in [-\pi, \pi[$, for instance:

$$\mathsf{V}(\vartheta_1, \ldots, \vartheta_n) = \sqrt{\mathsf{AMean}(\sin\vartheta_1, \ldots, \sin\vartheta_n)^2 + \mathsf{AMean}(\cos\vartheta_1, \ldots, \cos\vartheta_n)^2}$$

is quite often used in practice. The interested reader is referred to [381] for further references.

5.3 Consensus, inequality, and other measures

Somehow related to spread measures are numerical characteristics that originate from decision making, ecology, and economics.

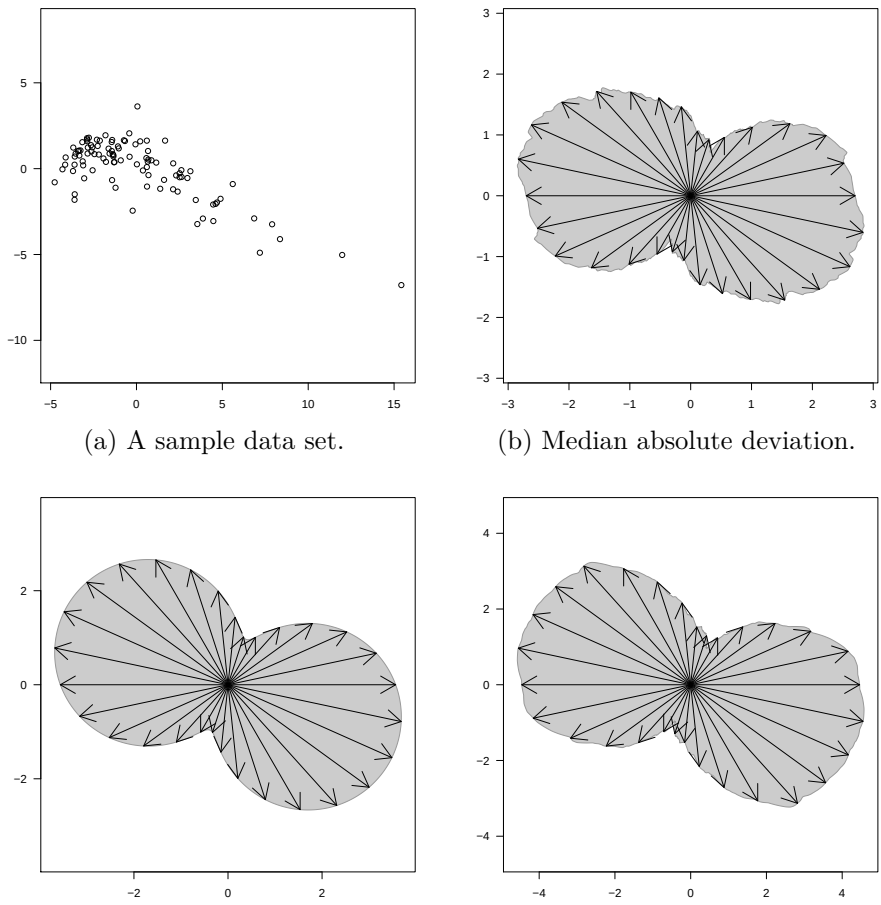
Measures of consensus and ecological evenness. Recently, Beliakov, Calvo, and James in [42] studied measures of decision makers' *consensus* that are based on Bonferroni means and fuzzy implications. They postulate that these should be functions like $\mathsf{C} : [0,1]^n \to [0,1]$ which fulfill at least the following properties:

— symmetry (unanimity),

— for all $x \in [0,1]$ it holds $\mathsf{C}(n * x) = 1$ (maximal consensus),

— $\mathsf{C}(\lfloor n/2 \rfloor * 0, \lceil n/2 \rceil * 1) = 0$ and $\mathsf{C}(\lfloor n/2 \rfloor * 1, \lceil n/2 \rceil * 0) = 0$ (minimal consensus),

— monotonicity with respect to the majority, i.e., for each $c \in [0,1]$ and $\mathbf{x}, \mathbf{y} \in [0,1]^{\lfloor n/2 \rfloor}$, if $|c - \mathbf{x}| \leq_{\lfloor n/2 \rfloor} |c - \mathbf{y}|$, then $\mathsf{C}(\lceil n/2 \rceil * c, \mathbf{x}) \geq \mathsf{C}(\lceil n/2 \rceil * c, \mathbf{y})$.



(a) A sample data set. (b) Median absolute deviation.

(c) Standard deviation. (d) Interquartile range.

Figure 5.2. An exemplary 2D data set together with one dimensional dispersion measures computed for its projections in every direction (represented as arrow lengths). Note that MAD and IQR give nonsmooth shapes.

Indices of social inequality and poverty. Economists find their interests in measures of social *inequality* (unevenness, poverty, etc.) For instance, Marques Pereira and others [14, 70, 218] (see, e.g., [283, 284] for a different setting) study poverty measures for nonnegative vectors defined as functions – among others – monotone with respect to the Lorenz majorization relation $\sqsubseteq_{\mathrm{L}}$, which is defined as $\mathbf{x} \sqsubseteq_{\mathrm{L}} \mathbf{y}$ if and only if $\mathsf{AMean}(\mathbf{x}) = \mathsf{AMean}(\mathbf{y})$ and $\mathsf{cumsum}(x_{(n)}, \ldots, x_{(1)}) \leq_n \mathsf{cumsum}(y_{(n)}, \ldots, y_{(1)})$. In particular, it is easily seen that $(n * \mathsf{AMean}(\mathbf{x})) \sqsubseteq_{\mathrm{L}} \mathbf{x}$ for all $\mathbf{x}$. On a side note, recall that absolute spread measures are given via a diff-based relation $\preccurlyeq_n$ and that cumsum can be conceived as a dual operation to diff.

Remark 5.12. Monotonicity with respect to $\sqsubseteq_{\mathrm{L}}$ is also called *Schur-convexity* in the literature. Interestingly, if $\mathbf{w}, \mathbf{v}$ are weighting vectors of the same lengths, then, see, e.g., [71]:

— for all $\mathbf{x} \in \mathbb{I}^n$, $\mathsf{OWA}_{\mathbf{w}}(\mathbf{x}) \leq \mathsf{OWA}_{\mathbf{v}}(\mathbf{x})$ if and only if $\mathsf{cumsum}(\mathbf{w}) \geq_n \mathsf{cumsum}(\mathbf{v})$,

— $\mathsf{OWA}_{\mathbf{w}}$ is Schur-convex if and only if $\mathbf{w}$ is ordered nondecreasingly.

Moreover, an exponential mean EMean_γ is Schur-convex, whenever $\gamma \geq 0$, see [70].

It turns out, see [47, 48], that social inequality measures can be related to ecological indices of *evenness* [390], which aim to capture how evenly species' populations are distributed over a geographical region, compare [100, 244, 391].

Entropy of discrete probability mass functions. A noteworthy characterization of measures of entropy or uncertainty of discrete probability mass functions (represented as numeric vectors in $[0,1]^n$ with elements summing up to 1) was proposed by Martín, Mayor, and Suñer in [348], compare also [403, 425] for axiomatizations on different kinds of domains. Such a class includes the Shannon entropy, $\mathsf{Entropy}(\mathbf{w}) = -\sum_{i=1}^n w_i \log w_i$, and alike, see also [299]. Here, monotonicity with respect to a partial order $\sqsubseteq_{\mathrm{D}}$ such that $\mathbf{w} \sqsubseteq_{\mathrm{D}} \mathbf{v}$ if and only if for all $i \in [n]$ $w_i \leq v_i \leq 1/n$ or $w_i \geq v_i \geq 1/n$ is considered useful.

Measures of shape of empirical distributions. We strongly believe that this short overview would have left the reader with a feeling of dissatisfaction if the two following measures of an input data vector's empirical distribution shape had not been considered.

A measure of *skewness* quantifies the degree of non-symmetry of an empirical distribution. A negative or positive skew is observed if the mass of the distribution is concentrated on the right or, respectively, left of the corresponding data histogram. In this case, we may consider, e.g.:

$$\mathrm{skewness}(\mathbf{x}) = \frac{\frac{1}{n}\sum_{i=1}^n (x_i - \mathsf{AMean}(\mathbf{x}))^3}{\left(\frac{1}{n-1}\sum_{i=1}^n (x_i - \mathsf{AMean}(\mathbf{x}))^2\right)^{3/2}}. \tag{5.15}$$

Notably, Liu, Parelius, and Singh in [324], compare also the work of Oja [378], study different types of symmetry of multidimensional data samples, such as spherical, elliptical, antipodal, or angular ones.

On the other hand, a measure of *kurtosis* (peakedness/flatness) shall be sensitive to the movement of the probability mass from the shoulders of a distribution to its center or tails, e.g.:

$$\mathrm{kurtosis}(\mathbf{x}) = \frac{\frac{1}{n}\sum_{i=1}^n (x_i - \mathsf{AMean}(\mathbf{x}))^4}{\left(\frac{1}{n}\sum_{i=1}^n (x_i - \mathsf{AMean}(\mathbf{x}))^2\right)^2} - 3. \tag{5.16}$$

Please note that for samples following a normal distribution skewness and kurtosis are – on average – equal to 0.

5.4 Impact functions for informetric data

Let us assume that $\mathbb{I} = [0, \infty]$ and that our universe of discourse consists of informetric strings as in Section 3.2. This time, however, we would like to compute a numerical characteristic of a given $\mathbf{x} \in \mathcal{S} = \{(x_1, x_2, \ldots, x_d) : d \in \mathbb{N}, (\forall i \in [d])\ x_i \in \mathbb{I}, x_1 \geq x_2 \geq \cdots \geq x_d\}$ such that it reflects both:

— the number of items (e.g., scientific articles, posts, software packages) produced by an abstract information resources producer (e.g., a scientist, StackOverflow user, software engineer) and

— the quality of individual products.

In the informetric (in particular, scientometric) literature it is widely accepted, see, e.g., [200, 214, 395, 396, 407, 473–475], that such an *impact function* $\mathsf{F} : \mathcal{S} \to [0, \infty]$ to be applied in the so-called Producers Assessment Problem (PAP) should at least be:

— $\sqsubseteq_\gamma$-nondecreasing (compare Section 3.1.2) and

— such that $\mathsf{F}(0) = 0$.

Note that $\sqsubseteq_\gamma$-nondecreasingness implies both monotonicity with respect to each component as well as the vector's size (arity), see [214] for a proof.

Remark 5.13. Note that, originally, many proposals for bibliometric indices assumed that we aggregate the number of papers' citations, i.e., sequences with elements in $\mathbb{N}_0$. Generally, however, the paper quality measures may be arbitrary real numbers, for example when citations are normalized according to the number of coauthors, paper's time of publication, quality of a journal, and so forth, see, e.g., [215].

Some of the notable examples of impact functions are as follows:

— Total number of product qualities:

$$\mathsf{Sum}(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i, \tag{5.17}$$

or, more generally, a weighted sum of elements of $\mathbf{x} \in \mathcal{S}$. This includes, e.g., "the total number of citations of the five most cited papers".

— The Hirsch h-index [249]:

$$\mathsf{H}(x_1, \ldots, x_n) = \max\{h \in [0 : n] : x_h \geq h\}, \tag{5.18}$$

with convention $x_0 = x_1$.

— The Kosmulski MaxProd-index [298]:

$$\mathsf{MP}(x_1,\ldots,x_n) = \max\left\{i \cdot x_i : i \in [n]\right\}. \tag{5.19}$$

This index is a particular case of the (projected) l_p-indices, $p \geq 1$, see [211].

— The Egghe g-index [180]:

$$\mathsf{G}(x_1,\ldots,x_n) = \max\left\{g \in [0:n] : \sum_{i=1}^{g} x_i \geq g^2\right\}, \tag{5.20}$$

with convention $\sum_{i=1}^{0} \cdots = 0$ and $x_{n+1} = x_{n+2} = \cdots = 0$.

— The Woeginger w-index [474]:

$$\mathsf{W}(x_1,\ldots,x_n) = \max\left\{w \in [0:n] : x_i \geq w - i + 1 \text{ for all } i \leq w\right\}. \tag{5.21}$$

The h- and w-index are generalized by, e.g., the class of r_p-indices, $p \geq 1$, see [211].

— The $h(2)$-index [297]:

$$\mathsf{H2}(x_1,\ldots,x_n) = \max\left\{h \in [0:n] : x_h \geq h^2\right\}. \tag{5.22}$$

Note that the $h(2)$-index is one of the many examples of very simple, direct modifications of the h-index. Many authors considered settings other than "h^2" on the right side of Equation (5.22), e.g., "αh" for some $\alpha > 0$ or "h^β", $\beta \geq 1$, see [7].

All the introduced impact functions are *zero-insensitive*, that is, for all $\mathbf{x} \in \mathcal{S}$ it holds $\mathsf{F}(\mathbf{x}) = \mathsf{F}(\mathbf{x}, 0)$. Moreover, the h-, w-, and $h(2)$-indices are symmetric minitive, see [204], and additionally the h-index is also maxitive and modular.

5.4.1 Impact functions generated by universal integrals

Let us study the connection between zero-insensitive impact functions and universal integrals, see Section 1.3.2. In this setting, with no loss in generality, we may assume that the vectors we characterize are padded with 0s and that they are elements in $\mathcal{S}_d = \{(x_1, x_2, \ldots, x_d) \in \mathbb{I}^d : x_1 \geq x_2 \geq \cdots \geq x_d\}$ for some fixed d. We shall need a transformation from the vector space $\mathcal{S}_d$ into the space $\mathcal{R}^{(\Omega,\mathcal{F})}$ for some $(\Omega, \mathcal{F})$. Although the most straightforward choice is of course the measurable space $(\mathbb{N}, 2^{\mathbb{N}})$, it is not necessarily the most convenient one. Thus, we fix the space to $(\mathbb{I}, \mathcal{B}(\mathbb{I}))$.

Given $\mathbf{x} \in \mathcal{S}_d$, let $\langle \mathbf{x} \rangle \in \mathcal{R}^{(\mathbb{I},\mathcal{B}(\mathbb{I}))}$ such that:

$$\langle \mathbf{x} \rangle(t) = x_{\lfloor t+1 \rfloor}, \quad t \in \mathbb{I}.$$

It is easily seen that $\langle \mathbf{x} \rangle$ is a nonincreasing step function with steps possible only in points from $\mathbb{N}$. As a matter of fact, $\langle \mathbf{x} \rangle$ is often called by bibliometricians the *citation function* for the vector $\mathbf{x}$.

Let us consider the family Φ of functions $\mathsf{F} : \mathcal{S}_d \to \mathbb{I}$ given by the equation:

$$\mathsf{F}(\mathbf{x}) = \eta\Big(\mathcal{I}\big(\mu, \langle\varphi(\mathbf{x})\rangle\big)\Big) \tag{5.23}$$

where:

— $\varphi : \mathcal{S}_d \to \mathcal{S}_d$ – a function nondecreasing in each variable, $\varphi(0, 0, \ldots, 0) = (0, 0, \ldots, 0)$,

— $\mu : \mathcal{B}(\mathbb{I}) \to [0, \infty]$ – a monotone measure,

— $\mathcal{I}$ – a universal integral on $\mathcal{M}^{(\mathbb{I},\mathcal{B}(\mathbb{I}))} \times \mathcal{R}^{(\mathbb{I},\mathcal{B}(\mathbb{I}))}$,

— $\eta : \mathbb{I} \to \mathbb{I}$ – an increasing function, $\eta(0) = 0$.

Noteworthily, Gagolewski and Mesiar in [216] provide an easy-to-use algorithm that may be used to compute the function given by Equation (5.23).

We have what follows, see [216].

Theorem 5.14. *Each function* F *given by Equation* (5.23) *is a zero-insensitive impact function.*

It is important to discuss the implications of choosing different φ, μ, $\mathcal{I}$, and η on the aggregation process. Please note that the φ function may be used, e.g., to normalize citation records, and often will be set by extending a function of one variable φ' to $\mathcal{S}$, that is $\varphi(\mathbf{x}) = (\varphi'(x_1), \varphi'(x_2), \ldots)$. Many classical (citation-based) bibliometric indices assume that $\varphi'(x) = \lfloor x \rfloor$ or $\varphi'(x) = x$. The η function may be used to "calibrate" the output values, especially if we would like to compare the values of different impact functions. On the other hand, the monotone measure μ shall in turn often be set to be the Lebesgue measure λ or some monotone transformation of λ.

Example 5.15. It is easily seen that:

— $\mathsf{Sum}(\mathbf{x}) = \mathrm{Ch}(\lambda, \langle\mathbf{x}\rangle)$, i.e., a Choquet integral, see Equation (1.29),

— $\mathsf{H}(\mathbf{x}) = \mathrm{Su}(\lambda, \langle\lfloor\mathbf{x}\rfloor\rangle) = \lfloor \mathrm{Su}(\lambda, \langle\mathbf{x}\rangle)\rfloor$ (Sugeno integral, Equation (1.31)), see also [450],

— $\mathsf{MP}(\mathbf{x}) = \mathrm{Sh}(\lambda, \langle\mathbf{x}\rangle)$ (Shilkret integral, Equation (1.30)).

Example 5.16. Let $\varphi = \mathrm{id}$, $\mathcal{I} = \mathrm{Ch}$, and $\eta = \mathrm{id}$.

— If $\mu = \lambda$, then we get of course $\mathcal{I}(\lambda, \langle\mathbf{x}\rangle) = \sum_i x_i$.

— For $\mu(A) = \lambda(A)^2$ (a convex transformation), we obtain $\mathcal{I}(\lambda^2, \langle\mathbf{x}\rangle) = \sum_i (i^2 - (i-1)^2) \cdot x_i = 1x_1 + 3x_2 + 5x_3 + 7x_5 + 9x_6 + \ldots$. Thus, we put higher weight for productivity here.

— If $\mu(A) = \sqrt{\lambda(A)}$ (a concave transformation), then $\mathcal{I}(\sqrt{\lambda}, \langle \mathbf{x} \rangle) = \sum_i (\sqrt{i} - \sqrt{i-1}) \cdot x_i \simeq 1.00x_1 + 0.41x_2 + 0.32x_3 + 0.27x_4 + 0.24x_5 + 0.21x_6 + \dots$. In consequence, the top-cited papers are of greater significance.

For instance, consider two vectors $\mathbf{y} = (60, 30, 10, 4, 0, 0, \dots)$ (higher quality) and $\mathbf{z} = (15, 13, 11, 11, 9, 8, 7, 7, 6, 5, 3, 3, 2, 1, 1, 1, 1, 0, 0, \dots)$ (higher productivity). We have $\mathcal{I}(\lambda, \langle \mathbf{y} \rangle) = \mathcal{I}(\lambda, \langle \mathbf{z} \rangle) = 104$, $\mathcal{I}(\lambda^2, \langle \mathbf{y} \rangle) \simeq 228 < \mathcal{I}(\lambda^2, \langle \mathbf{z} \rangle) \simeq 1050$, and $\mathcal{I}(\sqrt{\lambda}, \langle \mathbf{y} \rangle) \simeq 76.7 > \mathcal{I}(\sqrt{\lambda}, \langle \mathbf{z} \rangle) \simeq 36.9$

Example 5.17. Let $\mathcal{I} = \mathrm{Su}$, $\mu = \lambda$, $\eta = \mathrm{id}$. We know that by choosing $\varphi(\mathbf{x}) = \lfloor \mathbf{x} \rfloor$ we obtain the h-index, H. It is easily seen that, e.g., $\mathrm{Su}(\lambda, \langle \lfloor \sqrt{\mathbf{x}} \rfloor \rangle) = \mathsf{H2}(\mathbf{x})$. As we already indicated, many other Hirsch-based indices actually use simple transformations of the input vector, such as the one above. Moreover, by dropping the floor function we obtain the generalization of the h-index that is real-valued.

The φ function may be used, e.g., to change the impact of extremely high-cited publications, like when we choose $\varphi(\mathbf{x}) = \log(\mathbf{x} + 1)$.

Example 5.18. Consideration of more complex $\varphi : \mathcal{S} \to \mathcal{S}$ functions may lead us to other notable numerical characteristics. For example, the g- and w-index. Let $\mathsf{cummin}, \mathsf{cumsum} : \mathbb{I}^d \to \mathbb{I}^d$ denote the cumulative minimum and sum, respectively, i.e.:

$$\begin{aligned} \mathsf{cummin}(\mathbf{x}) &= (x_1, x_1 \wedge x_2, x_1 \wedge x_2 \wedge x_3, \dots), \\ \mathsf{cumsum}(\mathbf{x}) &= (x_1, x_1 + x_2, x_1 + x_2 + x_3, \dots). \end{aligned}$$

Given $\mathbf{x} \in \mathcal{S}_d$ it holds:

$$\mathsf{G}(\mathbf{x}) = \mathrm{Su}\left(\lambda, \left\langle \left\lfloor 0 \vee \mathsf{cummin}\big(\mathsf{cumsum}(\mathbf{x}) - (1^2, 2^2, \dots) + (1, 2, \dots)\big) \right\rfloor \right\rangle\right),$$

and:

$$\mathsf{W}(\mathbf{x}) = \mathrm{Su}\left(\lambda, \left\langle \left\lfloor \mathtt{cummin}\big(\mathbf{x} + (1, 2, \dots) - 1\big) \right\rfloor \right\rangle\right).$$

Example 5.19. Let $\mathcal{I} = \mathrm{Sh}$, $\mu = \lambda$, $\varphi = \mathrm{id}$. By setting $\eta = \mathrm{id}$ we of course get the MaxProd-index, MP. We may note, however, that the valuations generated by this index cannot be easily compared to that of the h-index. For example, we get $\mathsf{H}(n * n, 0, 0, \dots) = n$ and $\mathsf{MP}(n * n, 0, 0, \dots) = n^2$. Thus, by setting $\eta(x) = \sqrt{x}$ we may obtain the "calibrated" version of the MaxProd index.

Of course, integrals other than the classical Choquet, Sugeno, or Shilkret, may also lead to interesting indices.

5.4.2 Properties of impact functions

Apart from zero-insensitivity, here are some other properties of impact functions that can be useful in practice while aggregating vectors of varying lengths, see [107]:

— *F-insensitivity*, see [212, 474], see also "conservative productivity increment" in [367] and the notion of "stability" in [46], which holds if for all $\mathbf{x} \in \mathcal{S}$ and $0 \leq y \leq \mathsf{F}(\mathbf{x})$ we have $\mathsf{F}(\mathbf{x}, y) = \mathsf{F}(\mathbf{x})$,

— *F+sensitivity*, see [212, 474], see also "productivity responsiveness" in [367], that is for all $\mathbf{x} \in \mathcal{S}$ and $y > \mathsf{F}(\mathbf{x})$ we have $\mathsf{F}(\mathbf{x}, y) > \mathsf{F}(\mathbf{x})$,

— *multiplicative coherence*, compare [460], i.e., for all $\mathbf{x}, \mathbf{y} \in \mathcal{S}$ and $d \geq 1$ if $\mathsf{F}(\mathbf{x}) \leq \mathsf{F}(\mathbf{y})$, then $\mathsf{F}(d\mathbf{x}) \leq \mathsf{F}(d\mathbf{y})$,

— *additive coherence*, i.e., for all $\mathbf{x}, \mathbf{y} \in \mathcal{S}$ and $e \geq 0$ if $\mathsf{F}(\mathbf{x}) \leq \mathsf{F}(\mathbf{y})$, then $\mathsf{F}(\mathbf{x} + e) \leq \mathsf{F}(\mathbf{y} + e)$,

— *independence*, which was considered in [75], and states that the relative ranking of two producers should not change after an addition of products of the same quality; in other words, for all $\mathbf{x}, \mathbf{y} \in \mathcal{S}$ and $z \in \mathbb{I}$ it holds $\mathsf{F}(\mathbf{x}) \leq \mathsf{F}(\mathbf{y}) \Rightarrow \mathsf{F}(\mathbf{x}, z) \leq \mathsf{F}(\mathbf{y}, z)$,

— *consistency*, see [75], which considers joint output of consortia of producers: if a producer A is dominated by producer B, and C is dominated by D, then it is reasonable that A and C together (i.e., their combined outputs) shall be dominated by B and D; in other words, whenever for all $\mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}'$ such that $\mathsf{F}(\mathbf{x}) \leq \mathsf{F}(\mathbf{y})$ and $\mathsf{F}(\mathbf{x}') \leq \mathsf{F}(\mathbf{y}')$ it holds $\mathsf{F}(\mathbf{x}, \mathbf{x}') \leq \mathsf{F}(\mathbf{y}, \mathbf{y}')$.

Example 5.20. Let us consider the following impact functions:

— $\mathsf{Max}(\mathbf{x}) = x_1$ (sample maximum),

— $\mathsf{MaxN}(\mathbf{x}) = x_1 \wedge n$,

— $\mathsf{Q5}(\mathbf{x}) = x_5$ if $n \geq 5$ and 0 otherwise ($\sim$ the fifth quantile),

— $\mathsf{H}(\mathbf{x}) = \bigvee_{i=1}^{n} \lfloor x_i \rfloor \wedge i$ (the Hirsch index),

— $\widetilde{\mathsf{H}}(\mathbf{x}) = \bigvee_{i=1}^{n} x_i \wedge i$ (a real-valued Hirsch index),

— $\mathsf{H2}(\mathbf{x}) = \bigvee_{i=1}^{n} \lfloor \sqrt{x_i} \rfloor \wedge i$ (the $h^{(2)}$-index),

— $\widetilde{\mathsf{H2}}(\mathbf{x}) = \bigvee_{i=1}^{n} \sqrt{x_i} \wedge i$ (a real-valued $h^{(2)}$ index),

— $\mathsf{N}(\mathbf{x}) = n$ (sample length),

— $\mathsf{NP}(\mathbf{x}) = \sum_{i=1}^{n} \mathbf{1}(x_i > 0) = \bigvee_{i=1}^{n} \mathbf{1}(x_i > 0) b \wedge i$ (number of elements with non-zero quality).

All of these are symmetric minitive, maxitive, as well as modular. Table 5.1 summarizes which of the properties discussed in this section are fulfilled by the above functions. The function that obeys the greatest number of properties is the Max function.

Table 5.1. Exemplary impact functions and some properties they fulfill, see [107].

property name	Max	MaxN	Q5	H	$\widetilde{\mathsf{H}}$	H2	$\widetilde{\mathsf{H2}}$	N	NP	$\sum$
arity-monotonicity	●	●	●	●	●	●	●	●	●	9
continuity	●	●	●	○	●	○	●	●	○	6
zero-insensitivity	●	○	●	●	●	●	●	○	●	7
F-insensitivity	●	○	●	●	●	●	○	○	○	5
F+sensitivity	●	●	○	○	○	○	○	●	●	4
multiplicative coh.	●	○	●	○	○	○	○	●	●	4
additive coherent	●	○	○	○	○	○	○	●	○	2
independence	●	○	○	○	○	○	○	●	●	3
consistency	●	●	○	○	○	○	○	●	●	4
$\sum$	9	4	5	3	4	3	3	7	6	$\sum$

5.5 Characteristics of fusion functions

Numerical characteristics of fusion functions may give us a better insight into particular aspects of their behavior. Moreover, they can aid in selecting an aggregation function that best fits a practitioner's needs.

5.5.1 Orness and related measures

First we shall focus on idempotent fusion functions like $\mathsf{F} : \mathbb{I}^n \to \mathbb{I}$, see Chapter 1. Let us incorporate the traditional assumption that $\mathbb{I} = [0, 1]$. However, please note that the measures introduced in this section may be easily generalized to any $\mathbb{I}$ of finite width.

First let us note that the average value of $\mathsf{F}^{(n)}$ as defined in [230, Chapter 10], that is:

$$\text{average}(\mathsf{F}^{(n)}) = \int_{\mathbb{I}^n} \mathsf{F}^{(n)}(\mathbf{x})\, d\mathbf{x} \tag{5.24}$$

is nothing more than the expected value of $\mathsf{F}^{(n)}$ under the assumption that it is applied on a random vector uniformly distributed on $\mathbb{I}^n$.

Moreover, recall that in the class of idempotent aggregation functions, Min and Max are the least and the greatest fusion tools, respectively.

Lemma 5.21 ([171]). *For any n it holds:*

- $\text{average}(\mathsf{Min}^{(n)}) = \int_{\mathbb{I}^n} \mathsf{Min}^{(n)}(\mathbf{x})\, d\mathbf{x} = \frac{1}{n+1}$,
- $\text{average}(\mathsf{Max}^{(n)}) = \int_{\mathbb{I}^n} \mathsf{Max}^{(n)}(\mathbf{x})\, d\mathbf{x} = \frac{n}{n+1}$.

The orness measure for averaging functions was introduced by Dujmović in [172] under the name *disjunction degree*. This numerical characteristic aims to quantify how far – on average – a fusion function's value is from the least and the greatest averaging functions.

Definition 5.22. Let $\mathsf{F}^{(n)}$ be an idempotent aggregation function. Its degree of *orness* is given by:

$$\operatorname{orness}(\mathsf{F}^{(n)}) = \frac{\int_{\mathbb{I}^n} \mathsf{F}^{(n)}(\mathbf{x})\,d\mathbf{x} - \int_{\mathbb{I}^n} \mathsf{Min}^{(n)}(\mathbf{x})\,d\mathbf{x}}{\int_{\mathbb{I}^n} \mathsf{Max}^{(n)}(\mathbf{x})\,d\mathbf{x} - \int_{\mathbb{I}^n} \mathsf{Min}^{(n)}(\mathbf{x})\,d\mathbf{x}} \in [0,1]. \tag{5.25}$$

It is easily seen that $\operatorname{orness}(\mathsf{Min}^{(n)}) = 0$ and $\operatorname{orness}(\mathsf{Max}^{(n)}) = 1$. Additionally, $\operatorname{orness}(\mathsf{AMean}^{(n)}) = 0.5$. As noted in [49], exact orness values are known only for a few of the most notable fusion functions and arities, in particular:

— $\operatorname{orness}(\mathsf{QMean}^{(2)}) = \frac{1}{3}\left(1 + \frac{\log(1+\sqrt{2})}{\sqrt{2}}\right)$,

— $\operatorname{orness}(\mathsf{HMean}^{(2)}) = \frac{4}{3}(1 - \log 2)$,

— $\operatorname{orness}(\mathsf{GMean}^{(n)}) = \frac{n+1}{n-1}\left(\frac{n}{n+1}\right)^n - \frac{1}{n-1}$.

In other cases, computations may be performed numerically (via, e.g., numerical cubatures [53, 221] or Monte Carlo integration, especially if n is large). For instance:

```
qmean <- function(x) sqrt(mean(x^2)) # quadratic mean
# Compute orness measures for 1,2,3,4-ary quadratic means:
sapply(1:4, function(n) {
   cubature::adaptIntegrate(qmean,
      lowerLimit=rep(0, n),
      upperLimit=rep(1, n),
      tol=1e-12
   )
})

##               [,1]        [,2]        [,3]        [,4]
## integral      0.5         0.5410751   0.554598    0.5609498
## error         5.5511e-15  5.4056e-13  5.5459e-13  5.609497e-13
## fEvaluations  15          62509       2872617     103726263
```

Please note that the number of function evaluations increases drastically as n gets larger.

Remark 5.23. We can also consider the *andness* measure defined as:

$$\operatorname{andness}(\mathsf{F}^{(n)}) = 1 - \operatorname{orness}(\mathsf{F}^{(n)}). \tag{5.26}$$

A slightly different measure was introduced by Fernández Salido and Murakami in [185].

Definition 5.24. The *average orness* of an idempotent aggregation function $\mathsf{F}^{(n)}$ is given by:

$$\text{aveorness}(\mathsf{F}^{(n)}) = \int_{\mathbb{I}^n} \frac{\mathsf{F}^{(n)}(\mathbf{x}) - \mathsf{Min}^{(n)}(\mathbf{x})}{\mathsf{Max}^{(n)}(\mathbf{x}) - \mathsf{Min}^{(n)}(\mathbf{x})}\, d\mathbf{x}, \tag{5.27}$$

under the assumption that $0/0 = 0$.

Again, $\text{aveorness}(\mathsf{Min}^{(n)}) = 0$ and $\text{aveorness}(\mathsf{Max}^{(n)}) = 1$. Please note that the nature of the introduced measures of disjunctivity is such that they are based on the fact that F outputs a single numeric value. Therefore, they cannot be easily generalized to fusion functions like $\mathsf{F}^{(n)} : (\mathbb{I}^d)^n \to \mathbb{I}^d$. A possible idea to overcome this limitation would be to consider, e.g., a quite different measure based on a properly normalized expected Euclidean distance between the outputs generated by $\mathsf{F}^{(n)}$ and the boundary of the $\mathbb{I}^d$ set under the assumption that input data are independent and uniformly distributed on $\mathbb{I}^d$. Alternatively, taking into account the fact that if $X \sim \mathrm{U}(\mathbb{I}^d)$, then $\mathbb{E}\, X = (d*0.5)$, we may consider simply a mean squared error-like measure $\mathbb{E}\, \mathfrak{d}_2(\mathsf{F}^{(n)}(X_1, \ldots, X_n), (d * 0.5))$.

What is more, if $\mathsf{F}^{(n)}$ is a Lipschitz function, then its corresponding Lipschitz constant may also be used as its numerical characteristic.

5.5.2 Weighting vector's entropy

For classes of aggregation operators that are parametrized via a weighting vector – such as weighted arithmetic means or OWA operators – their orness measures are dependent solely on the distribution of weights.

Remark 5.25. The orness and average orness measures coincide in the case of OWA operators, see [185]. For any weighting vector $\mathbf{w}$ it holds that:

$$\text{orness}(\mathsf{OWA}_{\mathbf{w}}) = \text{aveorness}(\mathsf{OWA}_{\mathbf{w}}) = \sum_{i=1}^{n} \frac{n-i}{n-1} w_i.$$

Nevertheless, this is not the case for arbitrary fusion functions.

Therefore, such types of numerical characteristics may aid in choosing a particular fusion function, see, e.g., [186] for an example of fitting OWA operators to empirical data under the constraint that the orness measure is fixed to some pre-established value.

In the current setting, a weighting vector's *entropy*:

$$\mathsf{Entropy}(\mathbf{w}) = -\sum_{i=1}^{n} w_i \log w_i \tag{5.28}$$

can be conceived as a measure of the degree to which all the input data are used in the aggregation process, compare [49]. It is easily seen that the entropy is maximized for $\mathbf{w} = (n*(1/n))$. During the choice of a suitable OWA operator, it is not unusual to fix the desired orness and then maximize the weights' entropy. Moreover, we can note that, e.g., spread or related measures introduced above may be used for this purpose as well.

5.5.3 Breakdown points and values

An *outlier* is most often defined as an observation that is too distant from other data points and thus it is in some way suspicious. It may be present due to a measurement or data input error, or simply because we are analyzing a sample following a heavy-tailed distribution. Beckman and Cook in [28] note what follows:

> *The concern over outliers is old and undoubtedly dates back to the first attempt to base conclusions on a set of statistical data. Comments by Bernoulli (1777) indicate that the practice of discarding discordant observations was commonplace 200 years ago.*

The notion of a breakdown point of a fusion function as discussed in this monograph has been introduced by Donoho in [159]. It is meant to serve as a measure of a function's robustness to the presence of potential outliers. Its aim is to express "the smallest amount of contamination which can cause the estimator to give an arbitrarily bad answer".

Definition 5.26 ([159]). The *breakdown point* of a fusion function $\mathsf{F}^{(n)} : (\mathbb{R}^d)^n \to \mathbb{R}^d$, $d \geq 1$, at $\mathbf{X} \in (\mathbb{R}^d)^n$ is given by:

$$\varepsilon(\mathsf{F}^{(n)}, \mathbf{X}) = \min_{m \in [n]} \left\{ \frac{m}{n} : \sup_{\mathbf{Y}_m \in (\mathbb{R}^d)^n} \|\mathsf{F}^{(n)}(\mathbf{X}) - \mathsf{F}^{(n)}(\mathbf{Y}_m)\| = \infty \right\},$$

where the supremum is over all possible data sets $\mathbf{Y}_m$ obtained from $\mathbf{X}$ in such a way that exactly m points are replaced with arbitrary values.

Clearly, the higher the breakdown point, the more insensitive to outliers a fusion function is at a given point. As it is noted by Lopuhaä and Rousseeuw in [327], for most of the functions studied in the literature, ε does not depend on $\mathbf{X}$. In any case, one might be interested in quantifying the "global" *breakdown value*:

$$\text{breakval}(\mathsf{F}^{(n)}) = \inf_{\mathbf{X} \in (\mathbb{R}^d)^n} \varepsilon(\mathsf{F}^{(n)}, \mathbf{X}) \in \left[\frac{1}{n}, 1\right]. \tag{5.29}$$

Please note that for translation equivariant fusion functions we have that $\text{breakval}(\mathsf{F}^{(n)}) \leq 0.5$, see [159]. It turns out that we have what follows.

Table 5.2. Breakdown values of exemplary fusion functions.

fusion function	breakdown point	reference
centroid	$\frac{1}{n}$	—
componentwise median	$\frac{1}{2}$	[327], see also [27]
1-median	$\frac{1}{2}$	[327]
Tukey median	$\in \left[\frac{1}{d+1}, \frac{1}{3}\right]$	[8]
Oja median	$\frac{2}{n+2}$ for $d = 2$	[376]
Convex hull peeling median	$\leq \frac{n+d+1}{(n+2)(d+1)}$	[159, 432]

Theorem 5.27 ([27]). *The componentwise median,* $\mathsf{CwMedian}^{(n)}$*, is the only fusion function that is componentwise nondecreasing, translation and uniform-scale equivariant that has a 50% breakdown value.*

On a side note, it is worth stressing that this result is derived from the notion of monotonicity which is generally very rare in the computational statistics literature.

Example 5.28. Table 5.2 gives breakdown points for some of the fusion functions studied in this book, see also [8, 160]. We may note that, e.g., the componentwise arithmetic mean (centroid) is extremely sensitive to outliers.

Remark 5.29. Outliers are not the only types of data "contamination" that can be considered. Among others we have *missing values*, see [416], and *censored observations*, see [135, Section 8.5].

5.6 Characteristics of fuzzy numbers

Recall from Section 4.3 that a fuzzy number is a kind of fuzzy subset of the real line. Let us briefly review the numerical characteristics of such objects.

Measures of central tendency (defuzzifiers). Let us first mention the notion of the *expected interval* of a fuzzy number A, proposed by Dubois and Prade in [167]:

$$\mathrm{EI}(A) = \left[\int_0^1 A_L(\alpha)\,d\alpha, \int_0^1 A_U(\alpha)\,d\alpha\right]. \tag{5.30}$$

The midpoint of the expected interval is called the *expected value* of a fuzzy number. It is given by:

$$\mathrm{EV}(A) = \frac{\int_0^1 A_L(\alpha)\,d\alpha + \int_0^1 A_U(\alpha)\,d\alpha}{2}. \tag{5.31}$$

Sometimes a generalization of the expected value, called *weighted expected value*, is useful. For a given $w \in [0,1]$, it is defined as:

$$\mathrm{EV}_w(A) = (1-w)\int_0^1 A_L(\alpha)\,d\alpha + w\int_0^1 A_U(\alpha)\,d\alpha. \tag{5.32}$$

It is easily seen that $\mathrm{EV}_{0.5}(A) = \mathrm{EV}(A)$.

On the other hand, the *value* of A was defined by Delgado, Vila, and Voxman in [144], see also [103], as:

$$\mathrm{val}(A) = \int_0^1 \alpha\,(A_L(\alpha) + A_U(\alpha))\;d\alpha. \tag{5.33}$$

The α term may be replaced with a generic weighting function $\mathsf{w}(\alpha)$, most often such that $\int_0^1 \mathsf{w}(\alpha)\,d\alpha = 0.5$.

Please note that the expected value or value may be used to "defuzzify" A. The introduced measures are translation and scale equivariant.

Example 5.30. If A denotes a trapezoidal fuzzy number, $A = \mathrm{T}(s_1, s_2, s_3, s_4)$, then $\mathrm{EV}(A) = (s_1+s_2+s_3+s_4)/4$ and $\mathrm{val}(A) = (s_1+s_4)/6 + (s_2+s_3)/3$.

Measures of nonspecifity. Among notions of "nonspecifity" of a fuzzy number we find, among others, what follows. The *width* of A [112] is defined as the width of its expected interval, that is:

$$\mathrm{width}(A) = \int_0^1 A_U(\alpha)\,d\alpha - \int_0^1 A_L(\alpha)\,d\alpha. \tag{5.34}$$

The *ambiguity* of A [144] is defined as:

$$\mathrm{amb}(A) = \int_0^1 \alpha\,(A_U(\alpha) - A_L(\alpha))\;d\alpha. \tag{5.35}$$

Moreover, the *standard deviation* of A, as introduced by Carlsson and Fullér in [103], is given by:

$$\mathrm{sd}(A) = \sqrt{\frac{1}{2}\int_0^1 \alpha\,(A_U(\alpha) - A_L(\alpha))^2\;d\alpha}. \tag{5.36}$$

The three measures are translation invariant and scale equivariant. A defuzzifier together with a nonspecifity measure may be used to project a fuzzy number to a real interval.

Example 5.31. If A denotes a trapezoidal fuzzy number, $A = \mathrm{T}(s_1, s_2, s_3, s_4)$, then $\mathrm{width}(A) = (s_3+s_4)/2 - (s_1+s_2)/2$ and $\mathrm{amb}(A) = (s_3-s_2)/3 + (s_4-s_1)/6$. If A is a triangular fuzzy number (i.e., a trapezoidal one with $s_2 = s_3$) we additionally have $\mathrm{sd}(A) = (s_4-s_1)/\sqrt{24}$.

Measures of fuzziness. Closely related to nonspecifity characteristics are measures of *fuzziness* of arbitrary fuzzy sets, see, e.g., [418, 464, 490], which are often axiomatized as functions: (a) outputting value of 0 if and only if they are applied on a crisp set and (b) monotone with respect to a partial ordering relation $\sqsubseteq_{\mathrm{F}}$ such that $A \sqsubseteq_{\mathrm{F}} B$ (A is less fuzzy than B) if and only if $(0.5 \wedge \mu_A(x)) \leq (0.5 \wedge \mu_B(x))$ and $(0.5 \vee \mu_A(x)) \geq (0.5 \vee \mu_B(x))$ for all x. We see that the membership degree of 0.5 is considered as the most vague.

5.7 Checksums

Let us end this chapter – as well as the whole monograph – with a class of numerical characteristics which is quite different from the above measures: those which are supposed to be very difficult to study analytically.

A *checksum* function provides us with a way to verify data integrity that can be broken due to:

— errors in data transmission,

— cryptographic attacks,

— malware (malicious software, e.g., viruses, Trojan horses, backdoors) injection,

and so on. Such tools are related to hash functions and fingerprint algorithms, which are also used to map (perhaps uniquely) a (possibly) large and complex data set into a much simpler domain. However, their purpose is quite different than that of checksums – they aim to aid in efficient object dictionary look-up.

Most of the checksum functions studied in the literature assume that an input data stream consists of chunks of bit sequences of a fixed length, $\Sigma = \{0,1\}^d$ for, e.g., $d = 8$, 16, 32, or 64. They are incremental (compare Definition 1.121) functions: to compute their value only a single pass through a data stream is required. Checksum functions map the data into a set of binary sequences of fixed length d', e.g.:

— $d' = 32$ for the CRC-32 algorithm (which is based on cyclic codes introduced by Prange [393], see also [104, 389]),

— $d' = 128$ for the MD5 checksum (introduced in RFC1321[2]) by R. Rivest, compare also, e.g., [54]),

— $d' = 256$ in the case of the SHA-256 algorithm (which was developed by the National Security Agency (NSA), see, e.g., [358]).

Typically, we represent checksum routine outputs as character strings consisting of hexadecimal digits ($\mathtt{0}, \ldots, \mathtt{9}, \mathtt{a}, \ldots, \mathtt{f}$). However, please note that each bit

[2]See `https://tools.ietf.org/html/rfc1321`.

sequence may be mapped to an unsigned binary (base-2) number:

$$(b_{d-1}b_{d-2}\cdots b_0)_2 = \sum_{i=0}^{d-1} b_i 2^i. \tag{5.37}$$

Because of this, each checksum is an integer number and thus a function that computes it may be conceived as a kind of bit sequence numerical characteristic, $\mathsf{F} : \Sigma^* \to [0 : 2^{d'} - 1] \subseteq \mathbb{N}_0$.

Example 5.32. We have:

```
x <- "fusion␣functions" # input string
paste(charToRaw(x), collapse="") # hex sequence
## [1] "667573696f6e2066756e6374696f6e73"
digest::digest(x, "crc32")  # CRC-32 checksum
## [1] "659348f9"
digest::digest(x, "md5")    # MD5 checksum
## [1] "ecb11c3a8afdbfdec37956b4997fcf55"
```

By default, checksum algorithms involve the following operations on data chunks:

— bitwise NOT, AND, OR, and XOR (exclusive OR),

— rotate-no-carry (e.g., 00010111 → 10001011),

— right-logical-shift with 0-padding (e.g., 00010111 → *0*0001011),

— addition (modulo $2^{d'}$).

Algorithm 5.33. *To get a general intuition about how checksum algorithms look, here is a fragment of* C++ *code to compute CRC-32.*

```
/* COPYRIGHT (C) 1986 Gary S. Brown.  You may use
 * this program, or code or tables extracted from it,
 * as desired without restriction. */

static uint32_t crc32_tab[256] = {
   0x00000000, 0x77073096, 0xee0e612c, 0x990951ba,
   // ......., ..........., ..........., ...........,
   0xb40bbe37, 0xc30c8ea1, 0x5a05df1b, 0x2d02ef8d
};

uint32_t crc32(const uint8_t* buf, size_t n)
{
   uint32_t crc = 0 ^ ~0U;
   while (n--)
      crc = crc32_tab[(crc ^ *buf++) & 0xFF] ^ (crc >> 8);
   return crc ^ ~0U;
}
```

Here ^ *stands for bitwise XOR,* & *for AND,* ~ *for NOT, and* >> *for right-logical-shift. The data stream is read byte by byte.*

Desired properties of checksum functions like F, especially in cryptographic tasks, include (compare [124]):

— *preimage resistance* – given a checksum $\mathbf{y}$, it should be computationally infeasible to find a data stream $\mathbf{x}$ such that $\mathsf{F}(\mathbf{x}) = \mathbf{y}$,

— *collision resistance* – given a data stream $\mathbf{x}$, it should be computationally infeasible to find a data stream $\mathbf{x}' \neq \mathbf{x}$ such that $\mathsf{F}(\mathbf{x}) = \mathsf{F}(\mathbf{x}')$.

Generally, F itself should be relatively easy to compute but difficult to analyze and thus break (invert).

Another useful feature that F should possess is in full opposition to the Lipschitz continuity property: we would like a checksum to change drastically even for a very small perturbation in input streams. The dissimilarity degree can be expressed as, for instance:

— the Hamming distance in the case of base-2 representation of outputs,

— the Levenshtein distance for the character string (hexadecimal) form,

— absolute difference in the case of the numeric representation.

Example 5.34. Let us slightly modify a string from the previous example:

```
x <- "Fusion␣functions" # input string
paste(charToRaw(x), collapse="") # hex sequence
## [1] "467573696f6e2066756e6374696f6e73"
digest::digest(x, "crc32")  # CRC-32 checksum
## [1] "72ac3384"
digest::digest(x, "md5")    # MD5 checksum
## [1] "5c303067a54b75760940ba0d20ad01b7"
```

Please note that the checksums are very different. For instance, if CRC-32 checksums are interpreted as unsigned integers, the corresponding decimal numbers are equal to 1704151289 and 1923888004, respectively.

Appendix A

Listings

SOURCE codes of scripts or programs included in this book are licensed under the MIT license. The license permits code reuse within proprietary software provided that all copies of the software include the license terms and the copyright notice.

```
#include <algorithm>
// [[Rcpp::plugins("cpp11")]]

struct Comparer {
   const double* v;
   Comparer(const double* _v) { v = _v; }
   bool operator()(const int& i, const int& j) {
      // returns true if the first argument is less than
      // (i.e. is ordered before) the second.
      return (v[i] < v[j]);
   }
};

bool is_comonotonic(NumericVector x, NumericVector y) {
   int n = x.size();
   if (y.size() != n) stop("lengths␣of␣x␣and␣y␣differ");

   // recall that array elements in C++ are numbered from 0
   // let s = (0,1,...,n-1)
   std::vector<int> s(n); for (int i=0; i<n; ++i) s[i] = i;

   Comparer ltx(REAL(x));
   std::sort(s.begin(), s.end(), ltx);
   // now s is an ordering permutation of x

   Comparer lty(REAL(y));
   int i1 = 0;
   while (i1 < n) { /* now search for the longest subsequence
                       consisting of equal x's */
      int i2 = i1+1;
      while (i2 < n && x[s[i1]] == x[s[i2]]) ++i2;
      // sort the subsequence if necessary:
      if (i2-i1 > 1) std::sort(s.begin()+i1, s.begin()+i2, lty);
      // y[s[i1-1]]>y[s[i1]] => x and y are not comonotonic:
      if (i1 > 0 && y[s[i1-1]] > y[s[i1]]) return false;
      i1 = i2;
   }

   // as a by-product, (s[0]+1, s[1]+1, ..., s[n-1]+1)
   // is a permutation that orders both x and y
   return true;
}
```

Figure A.1. A C++ implementation of an $O(n \log n)$ algorithm to determine if two vectors of length n are comonotonic, see [207].

```
#' @param D a symmetric positive-semidefinite n*n matrix
#' @param c a numeric vector of length n
#' @param A an m*n numeric matrix
#' @param b a numeric vector of length m
#' @param r a character vector of length m
#'        with elements like <=, ==, or >=;
#'        specifies types of linear constraints
#' @param l a numeric vector of length n which gives
#'        lower bounds for corresponding x variables,
#'        -Inf gives no bound
#' @param u a numeric vector of length n which gives
#'        upper bounds for corresponding x variables,
#'        Inf gives no bound
#' @param c0 a single numeric value
#'
#' @return
#' A list with the following components:
#'  par: The best set of parameters, x, found;
#'  value: The value of the objective function at par;
#'  counts: The number of iterations that it took
#'          to solve the program;
#'  status: Solution status - 0 for optimal
cgal_qp_solver <- function(D, c, A, b, r=rep(">=", length(b)),
      l=rep(-Inf, length(c)), u=rep(Inf, length(c)), c0=0.0)
{
   stopifnot(is.numeric(D), is.finite(D), is.matrix(D))
   stopifnot(is.numeric(A), is.finite(A), is.matrix(A))
   stopifnot(is.numeric(c), is.finite(c))
   stopifnot(is.numeric(b), is.finite(b))
   stopifnot(is.character(r), r %in% c("<=", "==", ">="))
   stopifnot(is.numeric(l), !is.na(l) & !is.nan(l))
   stopifnot(is.numeric(u), !is.na(u) & !is.nan(u))
   stopifnot(is.numeric(c0), is.finite(c0))
   stopifnot(length(b) == nrow(A), length(r) == nrow(A))
   stopifnot(isSymmetric(D), ncol(D) == ncol(A))
   stopifnot(length(c) == nrow(D), length(c0) == 1)
   stopifnot(length(l) == nrow(D), length(u) == nrow(D))

   r <- match(r, c("<=", "==", ">="))-2 # values in {-1, 0, 1}
   fl <- is.finite(l) # which lower bounds for x are active
   fu <- is.finite(u) # which upper bounds for x are active

   .cgal_qp_solver(length(c), length(b), A, b, r,
      fl, l, fu, u, D, c, c0)
}
```

Figure A.2. An R interface to the CGAL [442] library quadratic programming solver, part I.

```
#include <CGAL/QP_functions.h>
#include <CGAL/MP_Float.h>
typedef CGAL::MP_Float ET;
typedef CGAL::Quadratic_program_solution<ET> Solution;
typedef CGAL::Quadratic_program_from_iterators<double**,
   double*, CGAL::Comparison_result*,
   int*, double*, int*, double*, double**, double*> Program;
// [[Rcpp::plugins("cpp11")]]
// [[Rcpp::export(".cgal_qp_solver")]]
List cgal_qp_solver(int n, int m, NumericMatrix A,
   NumericVector b, IntegerVector r, LogicalVector fl,
   NumericVector l, LogicalVector fu, NumericVector u,
   NumericMatrix D, NumericVector c, double c0)
{
  double *Aptr = REAL((SEXP)(A)), *Dptr = REAL((SEXP)(D));
  double **_A = new double*[n], **_D = new double*[n];
  for (int i=0; i<n; ++i) // ith column
  {   _A[i] = Aptr+i*m; _D[i] = Dptr+i*n; }
  CGAL::Comparison_result* _r = new CGAL::Comparison_result[m];
  for (int j=0; j<m; ++j)
     _r[j] = (r[j] < 0 ? CGAL::SMALLER
                  : (r[j] > 0 ? CGAL::LARGER : CGAL::EQUAL));
  List retval;
  Program qp(n, m, _A, REAL((SEXP)(b)), _r,
     (int*)LOGICAL((SEXP)(fl)), REAL((SEXP)(l)),
     (int*)LOGICAL((SEXP)(fu)), REAL((SEXP)(u)),
     _D, REAL((SEXP)(c)), c0);
  Solution s(CGAL::solve_quadratic_program(qp, ET()));
  // generate output solution:
  NumericVector solution(n);
  int i=0;
  for (auto it = s.variable_values_begin();
        it != s.variable_values_end(); ++it)
     solution[i++] = to_double(*it);
  retval = List::create(
     _("par") = solution,
     _("value") = to_double(s.objective_value()),
     _("counts") = s.number_of_iterations(),
     _("status") = (s.status() == CGAL::QP_OPTIMAL ? 0
                  : (s.status() == CGAL::QP_INFEASIBLE ? 1
                  : (s.status() == CGAL::QP_UNBOUNDED ? 2
                  : -1)))
  );
  delete [] _r; delete [] _A; delete [] _D;
  return retval;
}
```

Figure A.3. An R interface to the CGAL [442] library quadratic programming solver, part II.

```
fit_wam_L2_quadprog <- function(X, Y) {
   stopifnot(is.matrix(X), is.matrix(Y))
   n <- nrow(X); m <- ncol(X)
   stopifnot(1 == nrow(Y), m == ncol(Y))

   # Linear constraint (sum(w) == 1):
   A <- matrix(1, ncol=n, nrow=1)
   B <- 1

   # Objective function definition:
   D <- tcrossprod(X)      #    X %*% t(X)
   C <- -tcrossprod(X, Y) # - (X %*% t(Y))

   res <- cgal_qp_solver(D, C, A, B, r="==", l=rep(0, n))
   stopifnot(res$status == 0)
   res$par # return value
}
```

Figure A.4. R code for least squares fitting of weighted arithmetic mean's weights.

```
fit_wam_L1_linprog <- function(X, Y) {
   stopifnot(is.matrix(X), is.matrix(Y))
   n <- nrow(X); m <- ncol(X)
   stopifnot(1 == nrow(Y), m == ncol(Y))

   A <- rbind(
      cbind(t(X), -diag(m), diag(m)),
      c(rep(1, n), rep(0, 2*m))
   )
   B <- c(Y, 1)
   C <- c(rep(0, n), rep(1, 2*m))
   D <- matrix(0, nrow=n+2*m, ncol=n+2*m) # an LP problem
   res <- cgal_qp_solver(D, C, A, B, r=rep("==", nrow(A)),
                                     l=rep(0, n+2*m))
   stopifnot(res$status == 0)
   stopifnot(0 == max(apply(MARGIN=2, FUN=min,
      X=matrix(res$par[-(1:n)], nrow=2, byrow=TRUE))))
   res$par[1:n] # return value: first n parameters
}
```

Figure A.5. R code for least absolute deviation fitting of a weighted arithmetic mean's weights.

```
fit_wam_LInf_linprog <- function(X, Y) {
   stopifnot(is.matrix(X), is.matrix(Y))
   n <- nrow(X); m <- ncol(X)
   stopifnot(1 == nrow(Y), m == ncol(Y))
   A <- rbind(
      cbind(t(X), -1),
      cbind(t(X), +1),
      c(rep(1, n), 0)
   )
   B <- c(Y, Y, 1)
   C <- c(rep(0, n), 1)
   D <- matrix(0, nrow=n+1, ncol=n+1) # an LP problem
   res <- cgal_qp_solver(D, C, A, B,
      r=c(rep("<=", m), rep(">=", m), "=="),
      l=rep(0, n+1))
   stopifnot(res$status == 0)
   res$par[1:n] # return value: first n parameters
}
```

Figure A.6. R code for least Chebyshev metric fitting of a weighted arithmetic mean's weights.

```
fit_wqam_L2_optim <- function(X, Y, phi, phiInv, phiInvPrime) {
   stopifnot(is.matrix(X), is.matrix(Y))
   n <- nrow(X); m <- ncol(X)
   stopifnot(1 == nrow(Y), m == ncol(Y))
   stopifnot(is.function(phi),  # generator function
      is.function(phiInv),      # its inverse
      is.function(phiInvPrime)) # derivative of inverse
   phiX <- phi(X)
   w0 <- runif(n); w0 <- w0/sum(w0)
   lambda0 <- log(w0) # initial parameters
   E <- function(lambda) { # goodness-of-fit measure
      w <- exp(lambda)/sum(exp(lambda))
      sum((phiInv(t(w) %*% phiX)-Y)^2)
   }
   gradE <- function(lambda) { # its gradient
      w <- exp(lambda)/sum(exp(lambda))
      Z <- as.numeric(t(w) %*% phiX)
      2*w*(((phiInv(Z)-Y)*phiInvPrime(Z)) %*% (t(phiX) - Z))
   }
   res <- optim(lambda0, E, gradE, method="BFGS")
   stopifnot(res$convergence == 0)
   exp(res$par)/sum(exp(res$par)) # return value
}
```

Figure A.7. R code for least squares fitting of a weighted quasi-arithmetic mean's weights.

```
fit_wqam_L1_optim_approx <- function(X, Y,
      phi, phiInv, phiInvPrime, eps=1e-12) {
   stopifnot(is.matrix(X), is.matrix(Y))
   n <- nrow(X); m <- ncol(X)
   stopifnot(1 == nrow(Y), m == ncol(Y))
   stopifnot(is.function(phi),  # generator function
      is.function(phiInv),      # its inverse
      is.function(phiInvPrime)) # derivative of inverse
   stopifnot(is.numeric(eps), length(eps) == 1, eps > 0)

   phiX <- phi(X)
   w0 <- runif(n); w0 <- w0/sum(w0)
   lambda0 <- log(w0) # initial parameters

   E <- function(lambda) { # goodness-of-fit measure
      w <- exp(lambda)/sum(exp(lambda))
      e <- phiInv(t(w) %*% phiX)-Y
      sum(sqrt(e^2+eps^2))
   }

   gradE <- function(lambda) { # its gradient
      w <- exp(lambda)/sum(exp(lambda))
      Z <- as.numeric(t(w) %*% phiX)
      w*(
        ((phiInv(Z)-Y)*phiInvPrime(Z)/
           sqrt((phiInv(Z)-Y)^2 + eps^2)) %*%
        (t(phiX)-Z)
      )
   }

   res <- optim(lambda0, E, gradE, method="BFGS")
   stopifnot(res$convergence == 0)
   exp(res$par)/sum(exp(res$par)) # return value
}
```

Figure A.8. R code for approximate least absolute deviation fitting of a weighted quasi-arithmetic mean's weights.

```
fit_powmean_L2_optim <- function(X, Y, pmin=0.1, pmax=10) {
   stopifnot(is.matrix(X), is.matrix(Y))
   n <- nrow(X); m <- ncol(X)
   stopifnot(1 == nrow(Y), m == ncol(Y))

   p <- 1 # this will be a parameter shared by the 3 functions:
   phi          <- function(x)
      x^p
   phiInv       <- function(x)
      exp(log(x)/p) # x^(1/p)
   phiInvPrime <- function(x)
      exp((1-p)*log(x)/p)/p # (x^(1/p-1))/p
   envir_p <- new.env()
   envir_p[["p"]] <- p
   environment(phi) <- envir_p
   environment(phiInv) <- envir_p
   environment(phiInvPrime) <- envir_p

   E <- function(p) {
      assign("p", p, environment(phi)) # affects 3 functions
      w <- fit_wqam_L2_optim(X, Y, phi, phiInv, phiInvPrime)
      sum((as.numeric((t(X^p) %*% w)^(1/p))-Y)^2)
   }

   optimize(E, c(pmin, pmax))$minimum
}
```

Figure A.9. R code for determining best exponent p in a least squares error power mean fitting task; calls a function given in Figure A.7.

```
seb <- function(X) {
   stopifnot(is.numeric(X), is.matrix(X))
   n <- ncol(X)
   # QP solver in CGAL determines argmin_v 0.5 v^T D v + c^T v
   XtX <- crossprod(X) # (t(X) %*% X)
   D <- 2.0*XtX
   C <- -diag(XtX)
   A <- matrix(rep(1, n), ncol=n)
   B <- 1
   res <- cgal_qp_solver(D, C, A, B, r="==", l=rep(0, n))
   stopifnot(res$convergence == 0)
   v <- res$par
   as.numeric(tcrossprod(v, X)) # v %*% t(X) - return value
}
```

Figure A.10. An R implementation of a QP-based [220] Euclidean 1-center finder.

```
NumericMatrix rortho(int d) {
  if (d < 1) stop("d␣<␣1");
  NumericMatrix A(d, d); // resulting matrix
  NumericVector x(d);    // auxiliary vector

  /* --- Step 1. --- */
  double theta = Rf_runif(0.0, 2.0*M_PI); // U[0,2pi]
  double b = double(Rf_runif(0.0, 1.0)<0.5)*2.0-1.0; // U{-1,1}
  A(0,0) =    cos(theta);    A(0,1) =    sin(theta);
  A(1,0) = -b*sin(theta);    A(1,1) =  b*cos(theta);

  /* --- Step 2.--- */
  for (int i=3; i<=d; ++i) {
     /* --- Steps 2.1. and 2.2. together --- */
     double xnorm = 0.0;
     for (int j=0; j<i; ++j) {
        x[j] = Rf_rnorm(0.0, 1.0);
        xnorm += x[j]*x[j];
     } // non-normalized z
     xnorm = sqrt(xnorm);
     x[0] = 1.0-x[0]/xnorm;
     double xnorm2 = x[0]*x[0];
     for (int j=1; j<i; ++j) {
        x[j] = -x[j]/xnorm;
        xnorm2 += x[j]*x[j];
     } // non-normalized x
     xnorm2 = sqrt(xnorm2);
     for (int j=0; j<i; ++j) x[j] /= xnorm2;
     /* --- Step 2.3. --- */
     for (int k=i-1; k>0; --k)
        for (int j=i-1; j>0; --j)
           A(j,k) = A(j-1, k-1);
     for (int j=1; j<i; ++j) A(0,j) = A(j,0) = 0.0;
     A(0,0) = 1.0; // now previous A is extended
      for (int k=0; k<i; ++k) {
        double x2 = 0.0;
        for (int j=0; j<i; ++j)
           x2 += x[j]*A(j,k);   // t(x)*extA(.,k)
        for (int j=0; j<i; ++j)
           A(j,k) -= 2*x[j]*x2; // extA-2*x*above
     }
  }
  return A; // Step 3.
}
```

Figure A.11. A C++ implementation of Algorithm 2.18: Generation of a random orthogonal $d \times d$ matrix.

```
#include <algorithm>
#include <utility>
// [[Rcpp::plugins("cpp11")]]
NumericVector Weiszfeld1median(NumericMatrix X,
      NumericVector w, NumericVector y0, double eps=1.0e-9) {
   int d = X.nrow();
   int n = X.ncol();
   if (w.length()  != n) stop("w.length()␣␣!=␣n");
   if (y0.length() != d) stop("y0.length()␣!=␣d");

   NumericVector y_last(d); // a new vector
   NumericVector y_cur(Rcpp::clone(y0)); // a deep copy of y0

   double lasterr;
   do {
      std::swap(y_last, y_cur); // swaps underlying pointers
      for (int j=0; j<d; ++j)
         y_cur[j] = 0.0;
      double w_over_d_x_y = 0.0;
      for (int i=0; i<n; ++i) {
         double d_xi_y = 0.0;
         for (int j=0; j<d; ++j)
            d_xi_y += (X(j, i)-y_last[j])*(X(j, i)-y_last[j]);
         d_xi_y = sqrt(d_xi_y);
         if (d_xi_y <= eps) return y_last; /* Step 2.1. */
         double w_over_d_xi_y = w[i]/d_xi_y;
         w_over_d_x_y += w_over_d_xi_y;
         for (int j=0; j<d; ++j)
            y_cur[j] += w_over_d_xi_y*X(j, i);
      }

      lasterr = 0.0;
      for (int j=0; j<d; ++j) {
         y_cur[j] /= w_over_d_x_y;
         lasterr += (y_cur[j]-y_last[j])*(y_cur[j]-y_last[j]);
      }
   } while (lasterr > eps*eps); /* Step 2.3. */
   return y_cur;
}
```

Figure A.12. A C++ implementation of the Weiszfeld procedure, see Algorithm 2.50, for determining the weighted 1-median.

```
#include <unordered_map>
// [[Rcpp::plugins("cpp11")]]
IntegerVector median_hamming(IntegerMatrix X) {
   int n = X.ncol();
   int d = X.nrow();
   IntegerVector out(d);
   for (int i=0; i<d; ++i) {
      std::unordered_map<int, int> ht; // hasthable
      for (int j=0; j<n; ++j)
         ht[X(i,j)]++; /* count occurrences of each
            letter; ints are default-constructed as 0 */

      int max = 0, argmax = -1;
      for (auto it=ht.cbegin(); it != ht.cend(); ++it)
         if (max < (*it).second) { // find a most frequently
            max = (*it).second;    // occurring letter
            argmax = (*it).first;
         }
      out[i] = argmax;
   }
   return out;
}
```

Figure A.13. A C++ implementation of a procedure to determine a solution to Equation (2.40) – a median with respect to the Hamming distance.

```
// [[Rcpp::export]]
IntegerVector hamming_dist_max(IntegerMatrix Y,
      IntegerMatrix X) {
   int nx = X.ncol(), ny = Y.ncol(), d = Y.nrow();
   if (X.nrow() != d) stop("X.nrow() != Y.nrow()");
   IntegerVector out(ny);
   for (int i=0; i<ny; ++i) {
      int max_hamming = 0;
      for (int j=0; j<nx; ++j) {
         // Hamming distance between Y[,i] and X[,j]
         int h = 0;
         for (int k=0; k<d; ++k) h += (int)(Y(k,i) != X(k,j));
         if (h > max_hamming) max_hamming = h;
      }
      out[i] = max_hamming;
   }
   return out;
}
```

Figure A.14. A helper function used in Figure A.15; determines the maximal Hamming distance between each vector in `Y` and all vectors in `X`.

```
hamming_closest_ga <- function(X, k=length(X)*8, niter=2000,
      lambdaMutMult=0.001) {
   n <- ncol(X)
   d <- nrow(X)
   S <- unique(as.integer(X))
   # expected value of number of bits to mutate per iteration:
   lambdaMut <- max(1, k*d*lambdaMutMult)

   selection <- function(P, f) {
      p <- (d-f+1)^3 # max(f) == d
      p <- p/sum(p)
      P[,sample(k, replace=TRUE, size=2*k, prob=p)]
   }

   crossover <- function(P2) { # uniform crossover
      P <- P2[,1:k]
      for (i in 1:k) {
         b <- sample(d, d/2)
         P[b, i] <- P2[b, i+k]
      }
      P
   }

   mutation <- function(P) {
      m <- sample(length(P), min(k*d, rpois(1, lambdaMut)))
      P[m] <- sample(S, length(m), replace=TRUE)
      P
   }

   # initial population: points in X and random ones (mixed):
   P <- matrix(nrow=d, sample(S, k*d, replace=TRUE))
   P[,sample(k, min(n, k))] <- X[,sample(n, min(n, k))]

   # store the best solution so far:
   f <- hamming_dist_max(P, X)
   bestP <- t(unique(t(P[,f==min(f)])))
   bestF <- min(f)

   for (i in 1:niter) {
      P <- mutation(crossover(selection(P, f)))
      f <- hamming_dist_max(P, X)
      if (bestF > min(f)) { # we got a better solution
         bestP <- t(unique(t(P[,f==min(f)])))
         bestF <- min(f)
      }
   }
   bestP # return value
}
```

Figure A.15. An R implementation of a genetic algorithm-based approximate solution to the closest vector with respect to the Hamming distance finding problem.

```
double dpr2_dist(List X, NumericVector y,
                 int dy, double p, double r) {
   int n = X.size();
   double dist = 0.0;

   for (int i=0; i<n; ++i) {
      NumericVector x(X[i]);
      int dx = x.size();
      int min_dx_dy = std::min(dx, dy);

      for (int j=0; j<min_dx_dy;  ++j)
         dist += (x[j]-y[j])*(x[j]-y[j]);

      for (int j=min_dx_dy; j<dx; ++j)
         dist += x[j]*x[j];

      for (int j=min_dx_dy; j<dy; ++j)
         dist += y[j]*y[j];

      dist += p*abs(pow(dx, r)-pow(dy, r));
   }

   return dist;
}
```

Figure A.16. A C++ implementation of to compute the sum of $\mathfrak{d}^2_{p,r}$ penalty functions, see Equation (3.5), between the first `dy` observations in a vector `y` and each vector in `X`.

```
#include <deque>
// [[Rcpp::plugins("cpp11")]]
NumericVector dpr2_centroid(List X, double p, double r) {
   int l=X.size();
   int d=calc_max_vector_length(X);

   NumericVector xtilde(d);
   for (auto it=X.begin(); it != X.end(); ++it) {
      NumericVector x(*it);
      int dx = x.size();
      for (int j=0; j<dx; ++j) xtilde[j] += x[j];
   }

   // a linked list (a stack):
   std::deque< std::pair<int, int> > part;
   NumericVector y(d);
   NumericVector best_y=NumericVector(0);
   double best_dist=INFINITY;

   for (int n=1; n<=d; ++n) {
      // C++ arrays use 0-based indices
      part.push_front( std::pair<int, int>(n-1, n-1) );
      y[n-1]=xtilde[n-1]/l;
      auto it=part.begin();
      while (it+1!=part.end() &&
            y[(*it).first] > y[(*(it+1)).second]) {
         // merge:
         int p1=(*it).second-(*it).first+1;
         int p2=(*(it+1)).second-(*(it+1)).first+1;
         y[(*it).second] = (y[(*it).second]*p1+
            y[(*(it+1)).second]*p2)/(p1+p2);
         for (int j=(*it).second-1; j>=(*(it+1)).first; --j)
            y[j]=y[(*it).second];
         (*(it+1)).second=(*it).second;
         // erase current it and move forward (pop stack)
         it=part.erase(it);
      }
      double cur_dist=dpr2_dist(X, y, n, p, r);
      if (cur_dist<best_dist) {
         best_dist=cur_dist;
         best_y=NumericVector(y.begin(), y.begin()+n);
      }
   }
   return best_y;
}
```

Figure A.17. A C++ implementation of a function to compute centroid-like fusion function for informetric data given by Equation (3.6); a function from Figure A.16 is called; see [106].

```
int levenshtein_smallmem(int* s1, int* s2, int n1, int n2) {
   if (n1 < n2) {
      std::swap(s1, s2); // pointer swap
      std::swap(n1, n2);
   }

   int* v_cur = new int[n2+1];
   int* v_last = new int[n2+1]; // n2 <= n1
   for (int j=0; j<=n2; ++j)
      v_cur[j] = j;

   for (int i=1; i<=n1; ++i) {
      std::swap(v_last, v_cur); // pointer swap
      v_cur[0] = i;
      for (int j=1; j<=n2; ++j) {
         v_cur[j] = std::min(std::min(
               v_last[j-1]+(int)(s1[i-1]!=s2[j-1]),
               v_cur[j-1]+1),
               v_last[j]+1);
      }
   }

   int ret = v_cur[n2];
   delete [] v_cur;
   delete [] v_last;
   return ret;
}


// [[Rcpp::export]]
int levenshtein_smallmem(IntegerVector s1, IntegerVector s2) {
   // Rcpp interface to the above function
   return levenshtein_smallmem(INTEGER(s1), INTEGER(s2),
      LENGTH(s1), LENGTH(s2));
}
```

Figure A.18. A memory-efficient C++ implementation of a Wagner-Fisher version [458] of the Levenshtein distance computation algorithm.

```
// [[Rcpp::export]]
IntegerVector Levenshtein_centroid2(IntegerVector s1,
      IntegerVector s2) {
   int n1 = s1.size(), n2 = s2.size();
   NumericMatrix D(n1+1, n2+1);
   IntegerMatrix T(n1+1, n2+1);
   for (int i=1; i<=n1; ++i) { // deletion
      D(i,0) = D(i-1,0)+1; T(i,0) = 4;
   }
   for (int j=1; j<=n2; ++j) { // insertion
      D(0,j) = D(0,j-1)+1; T(0,j) = 2;
   }
   for (int i=1; i<=n1; ++i)
      for (int j=1; j<=n2; ++j) {
         T(i,j) = 0;
         if (s1[i-1]==s2[j-1])
            D(i,j) = D(i-1,j-1);       // no change
         else {
            double m1 = D(i-1,j-1)+1;        // sub
            double m2 = D(i,j-1)+1;          // ins
            double m3 = D(i-1,j)+1;          // del
            D(i,j) = std::min(std::min(m1, m2), m3);
            if (D(i,j) == m1) T(i,j) |= 1;
            if (D(i,j) == m2) T(i,j) |= 2;
            if (D(i,j) == m3) T(i,j) |= 4;
         }
      }
   int maxd = (int)(D(n1, n2)*0.5);
   if (maxd <= 0) return s1;
   std::list<int> l1(s1.begin(), s1.end());
   auto it1 = l1.end(); --it1;
   auto it2 = s2.end(); --it2;
   int x = n1, y = n2;
   for (int curd=0; curd < maxd; ) {
      curd += (int)(T(x,y) != 0);
      if (T(x,y) == 0) {                          // no change
         x--; y--; --it1; --it2;
      } else if (T(x,y) & 1) {                          // sub
         x--; y--; (*(it1--)) = (*(it2--));
      } else if ((T(x, y) & 2) && ((!(T(x,y)&4))
            || ((int)l1.size() < std::max(n1,n2)))) {  // ins
         y--; it1 = l1.insert(++it1, *(it2--)); --it1;
      } else {                                          // del
         x--; it1 = l1.erase(it1); --it1;
      }
   }
   return IntegerVector(l1.begin(), l1.end());
}
```

Figure A.19. A C++ implementation of an algorithm to compute the Levenshtein distance-based centroid of two strings.

```
#include <algorithm>

struct Comparer {
   // just like the comparer for the comonotonicity algorithm
   const int* v;
   Comparer(const int* _v) { v = _v; }
   bool operator()(const int& i, const int& j) const {
      return v[i] < v[j];
   }
};


// [[Rcpp::export]]
double dinudist(IntegerVector x, IntegerVector y) {
   int nx = x.size(), ny = y.size();

   // ordering permutation of x:
   std::vector<int> ox(nx);
   for (int i=0; i<nx; ++i) ox[i] = i;
   std::stable_sort(ox.begin(), ox.end(),
      Comparer(INTEGER(x)));

   // ordering permutation of y:
   std::vector<int> oy(ny);
   for (int i=0; i<ny; ++i) oy[i] = i;
   std::stable_sort(oy.begin(), oy.end(),
      Comparer(INTEGER(y)));

   double d = 0.0;
   int ix = 0, iy = 0;
   while (ix < nx && iy < ny) {
      if (x[ox[ix]] == y[oy[iy]])
         d += std::abs((ox[ix++]+1) - (oy[iy++]+1));
      else if (x[ox[ix]] < y[oy[iy]])
         d += std::abs((ox[ix++]+1) - 0);
      else
         d += std::abs(0 - (oy[iy++]+1));
   }
   while (ix < nx)
      d += std::abs((ox[ix++]+1) - 0);
   while (iy < ny)
      d += std::abs(0 - (oy[iy++]+1));

   return d;
}
```

Figure A.20. A C++ implementation of an algorithm to compute the Dinu rank distance.

```
struct NNItem {
   size_t index;
   double dist;

   NNItem(size_t index, double dist) :
      index(index), dist(dist) {}

   NNItem() :
      index(SIZE_MAX), dist(-INFINITY) {}

   inline bool operator<( const NNItem& o ) const {
      return dist < o.dist;
   }
};

class Distance { // abstract class, must be overloaded
private:
   size_t n;
   virtual double compute(size_t v1, size_t v2) = 0;

public:
   Distance(Function distance, RObject objects);
   inline size_t getObjectCount() { return n; }
   inline double operator()(size_t v1, size_t v2) {
      return compute(v1, v2);
   }
};

double sumd_nn(Distance* dist, size_t i, size_t nntry,
    std::priority_queue<NNItem>& queue, double limit=INFINITY)
{
  double totd = 0.0;
  for (size_t j=0; j<dist->getObjectCount(); ++j) {
    if (i == j) continue;
    double curd = (*dist)(i, j);
    if (queue.empty())
       queue.push( NNItem(j, curd) );
    else if (curd <= queue.top().dist) {
       if (queue.size() >= nntry && curd < queue.top().dist) {
          double oldtop = queue.top().dist;
          while (!queue.empty() && queue.top().dist == oldtop)
             queue.pop();
       }
       queue.push( NNItem(j, curd) );
    }
    totd += curd;
    if (totd >= limit) return INFINITY;
  }
  return totd;
}
```

Figure A.21. Approximate medoid search in an arbitrary finite semimetric space, part I.

```
// [[Rcpp::export]]
RObject medoid_approx(Function distance, RObject objects,
      int iters=15, int nntry=5)
{
  RObject result(R_NilValue);
  Distance* dist = new Distance(distance, objects);
  size_t n = dist->getObjectCount();
  std::vector<bool> active(n, true);
  size_t besti_overall = -1;
  double bestd_overall = INFINITY;
  for (size_t r=0; r<(size_t)iters; ++r) {
      std::priority_queue<NNItem> queue;
      size_t besti = (size_t)(unif_rand()*n);
      if (!active[besti]) continue;
      active[besti] = false;
      double bestd = sumd_nn(dist, besti, nntry, queue);
      std::priority_queue<NNItem> bestqueue;
      bool change = true;
      while (change) {
         change = false;
         while (!queue.empty()) {
            NNItem nncur = queue.top();
            queue.pop();
            size_t curi = nncur.index;
            if (!active[curi]) continue;
            active[curi] = false;
            std::priority_queue<NNItem> curqueue;
            double curd = sumd_nn(dist, curi, nntry,
                                  curqueue, bestd);
            if (curd < bestd) {
              change = true;
              bestd = curd;
              besti = curi;
              bestqueue = curqueue;
            }
         }
         queue = bestqueue;
      }
      if (bestd < bestd_overall) {
         bestd_overall = bestd;
         besti_overall = besti;
      }
  }
  result = besti_overall+1;
  if (dist) delete dist;
  return result;
}
```

Figure A.22. Approximate medoid search in an arbitrary finite semimetric space, part II.

References

[1] Abellanas, M., Claverol, M., and Hurtado, F. Point set stratification and Delaunay depth. *Computational Statistics & Data Analysis 51* (2007), 2513–2530.

[2] Abreu, J., and Rico-Juan, J. A new iterative algorithm for computing a quality approximate median of strings based on edit operations. *Pattern Recognition Letters 36* (2014), 74–80.

[3] Aczel, A. *Complete Business Statistics*. Irvin, 1996.

[4] Aczél, J. On mean values. *Bulletin of the American Mathematical Society 54*, 4 (1948), 392–400.

[5] Aggarwal, C. C., Hinneburg, A., and Keim, D. A. On the surprising behavior of distance metrics in high dimensional space. *Lecture Notes in Computer Science 1973* (2001), 420–434.

[6] Aho, A., Garey, M., and Ullman, J. The transitive reduction of a directed graph. *SIAM Journal on Computing 1*, 2 (1972), 131–137.

[7] Alonso, S., Cabrerizo, F. J., Herrera-Viedma, E., and Herrera, F. *h*-index: A review focused on its variants, computation and standardization for different scientific fields. *Journal of Informetrics 3* (2009), 273–289.

[8] Aloupis, G. Geometric measures of data depth. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* (2006), pp. 147–158.

[9] Aloupis, G., Cortés, C., Gómez, F., Soss, M., and Toussaint, G. Lower bounds for computing statistical depth. *Computational Statistics & Data Analysis 40* (2002), 223–229.

[10] Aloupis, G., Langerman, S., Soss, M., and Toussaint, G. Algorithms for bivariate medians and a Fermat-Torricelli problem for lines. *Computational Geometry: Theory and Applications 26*, 1 (2003), 69–79.

[11] Aloupis, G., and McLeish, E. A lower bound for computing Oja depth. *Information Processing Letters 96* (2005), 151–153.

[12] Anderson, E., et al. *LAPACK Users' Guide*, 1999. SIAM. Available on-line at http://www.netlib.org/lapack/lug/lapack_lug.html.

[13] Angelov, P., and Yager, R. R. Density-based averaging – A new operator for data fusion. *Information Sciences 222* (2013), 163–174.

[14] Aristondo, O., García-Lapresta, J., Lasso de la Vega, C., and Marques Pereira, R. Classical inequality indices, welfare and illfare functions, and the dual decomposition. *Fuzzy Sets and Systems 228* (2013), 114–136.

[15] Arrow, K. J. *Social Choice and Individual Values*. Yale University Press, New Haven, 1963.

[16] Atanassov, K. *Intuitionistic Fuzzy Sets*. Physica-Verlag, Heidelberg, New York, 1999.

[17] Atanassov, K. T. Intuitionistic fuzzy sets. *Fuzzy Sets and Systems 20* (1986), 87–96.

[18] Baczyński, M., and Jayaram, B. *Fuzzy implications*. Springer-Verlag, Berlin, 2008.

[19] Baczyński, M., and Jayaram, B. (S, N)- and R-implications: A state-of-the-art survey. *Fuzzy Sets and Systems 159*, 14 (2008), 1836–1859.

[20] Bahlmann, C. Directional features in online handwriting recognition. *Pattern Recognition 39*, 1 (2006), 115–125.

[21] Ban, A. I., and Coroianu, L. Simplifying the search for effective ranking of fuzzy numbers. *IEEE Transactions on Fuzzy Systems 23* (2015), 327–339.

[22] Ban, A. I., Coroianu, L., and Grzegorzewski, P. Trapezoidal approximation and aggregation. *Fuzzy Sets and Systems 177*, 1 (2011), 45–59.

[23] Ban, A. I., Coroianu, L., and Grzegorzewski, P. A fixed-shape fuzzy median of a fuzzy sample. In *Proc. EUSFLAT'13* (2013), Atlantis Press, pp. 215–222.

[24] Bargiela, A., and Pedrycz, W. *Granular Computing: An Introduction*. Kluwer Academic Publishers, Boston, MA, 2003.

[25] Bartoszuk, M., and Gagolewski, M. A fuzzy R code similarity detection algorithm. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems, Part III* (2014), A. Laurent et al., Eds., vol. 444, Springer, pp. 21–30.

[26] Bartoszuk, M., and Gagolewski, M. Detecting similarity of R functions via a fusion of multiple heuristic methods. In *Proc. IFSA/Eusflat 2015* (2015), J. Alonso, H. Bustince, and M. Reformat, Eds., Atlantic Press, pp. 419–426.

[27] Bassett, Jr., G. W. Equivariant, monotonic, 50% breakdown estimators. *The American Statistician 45*, 2 (1991), 135–137.

[28] Beckman, R., and Cook, R. Outlier.....s. *Technometrics 25*, 2 (1983), 119–149.

[29] Bedall, F. K., and Zimmermann, H. Algorithm AS 143: The Mediancentre. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 28*, 3 (1979), 325–328.

[30] Bednarek, A. An extension of Light's associativity test. *American Mathematical Monthly 75*, 5 (1968), 531–532.

[31] Beliakov, G. Shape preserving approximation using least squares splines.

Approximation Theory and its Applications 16, 4 (2000), 80–98.

[32] Beliakov, G. Monotone approximation of aggregation operators using least squares splines. *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems 10* (2002), 659–676.

[33] Beliakov, G. How to build aggregation operators from data. *International Journal of Intelligent Systems 18* (2003), 903–923.

[34] Beliakov, G. Learning weights in the generalized OWA operators. *Fuzzy Optimization and Decision Making 4* (2005), 119–130.

[35] Beliakov, G. Monotonicity preserving approximation of multivariate scattered data. *BIT Numerical Mathematics 45* (2005), 653–677.

[36] Beliakov, G. Construction of aggregation operators for automated decision making via optimal interpolation and global optimization. *Journal of Industrial and Management Optimization 3*, 2 (2007), 193–208.

[37] Beliakov, G. Construction of aggregation functions from data using linear programming. *Fuzzy Sets and Systems 160* (2009), 65–75.

[38] Beliakov, G. Fast computation of trimmed means. *Journal of Statistical Software 39* (2011), Code snippet 2.

[39] Beliakov, G., Bustince, H., and Calvo, T. *A Practical Guide to Averaging Functions*. Springer, 2016.

[40] Beliakov, G., Bustince, H., James, S., Calvo, T., and Fernandez, J. Aggregation for Atanassov's intuitionistic and interval valued fuzzy sets: The median operator. *IEEE Transactions on Fuzzy Systems 20* (2011), 487–498.

[41] Beliakov, G., Calvo, T., and James, S. On penalty-based aggregation functions and consensus. In *Consensual Processes, STUDFUZZ 267* (2011), E. Herrera-Viedma et al., Eds., pp. 23–40.

[42] Beliakov, G., Calvo, T., and James, S. Consensus measures constructed from aggregation functions and fuzzy implications. *Knowledge-Based Systems 55* (2014), 1–8.

[43] Beliakov, G., Calvo, T., and Wilkin, T. Three types of monotonicity of averaging functions. *Knowledge-Based Systems 72* (2014), 114–122.

[44] Beliakov, G., Calvo, T., and Wilkin, T. On the weak monotonicity of Gini means and other mixture functions. *Information Sciences 300* (2015), 70–84.

[45] Beliakov, G., and James, S. Using linear programming for weights identification of generalized Bonferroni means in R. *Lecture Notes in Computer Science 7647* (2012), 35–44.

[46] Beliakov, G., and James, S. Stability of weighted penalty-based aggregation functions. *Fuzzy Sets and Systems 226*, 1 (2013), 1–18.

[47] Beliakov, G., and James, S. Unifying approaches to consensus across different preference representations. *Applied Soft Computing 35* (2015), 888–897.

[48] Beliakov, G., James, S., and Nimmo, D. Can indices of ecological evenness be used to measure consensus? In *Proc. IEEE Intl. Conf. Fuzzy Systems'15* (Beijing, China, 2014), pp. 1–8.

[49] Beliakov, G., Pradera, A., and Calvo, T. *Aggregation functions: A*

guide for practitioners. Springer-Verlag, 2007.

[50] Beliakov, G., and Warren, J. Appropriate choice of aggregation operators in fuzzy decision support systems. *IEEE Transactions on fuzzy systems 9*, 6 (2001), 773–784.

[51] Beliakov, G., and Wilkin, T. On some properties of weighted averaging with variable weights. *Information Sciences 281* (2014), 1–7.

[52] Benjamini, Y., and Hochberg, Y. Controlling False Discovery Rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B 57*, 1 (1995), 289–300.

[53] Berntsen, J., Espelid, T., and Genz, A. An adaptive algorithm for the approximate calculation of multiple integrals. *ACM Transactions on Mathematical Software 17*, 4 (1991), 437–451.

[54] Berson, T. A. Differential cryptanalysis mod 2^{32} with applications to MD5. *Lecture Notes in Computer Science 658* (1993), 71–80.

[55] Bezdek, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer, 1981.

[56] Bickel, P., and Lehmann, E. Descriptive statistics for nonparametric models. I. Introduction. II. Location. *The Annals of Statistics 3* (1975), 1039–1069.

[57] Bickel, P., and Lehmann, E. Descriptive statistics for nonparametric models. III. Dispersion. *The Annals of Statistics 4*, 6 (1975), 1139–1158.

[58] Bickel, P., and Lehmann, E. Descriptive statistics for nonparametric models. IV. Spread. In *Contributions to Statistics*, A. Jureckova, Ed. Academia, Prague, 1975, pp. 33–40.

[59] Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., and Fienberg, S. Adaptive name matching in information integration. *IEEE Intelligent Systems 18*, 5 (2003), 16–23.

[60] Bille, P. A survey on tree edit distance and related problems. *Theoretical Computer Science 337*, 1–3 (2005), 217–239.

[61] Billingsley, P. *Probability and Measure*. Wiley, 1979.

[62] Birkhoff, G. *Lattice Theory*. American Mathematical Society, Providence, RI, 1967.

[63] Bisschop, J., et al. *AIMMS Optimization Modeling*. Paragon Decision Technology, 2012.

[64] Bloch, I. Information combination operators for data fusion: A comparative review with classification. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans 26*, 1 (1996), 52–67.

[65] Bloomfield, P., and Steiger, W. L. *Least Absolute Deviations. Theory, applications, and algorithms*. Birkhäuser, Boston, Basel, Stuttgart, 1983.

[66] Blum, M., Floyd, R. W., Pratt, V., Rives, R. L., and Tarjan, R. E. Time bounds for selection. *Journal of Computer and System Sciences 7*, 4 (1973), 448–460.

[67] Boomsma, W., Mardia, K., Taylor, C., Ferkinghoff-Borg, J., Krogh, A., and Hamelryck, T. A generative, probabilistic model of local protein structure. *Proceedings of the National Academy of Sciences*

105, 26 (2008), 8932–8937.

[68] Borovskikh, Y. V. Nonuniform estimation of rate of convergence for L-statistics. *Ukrainian Mathematical Journal 33*, 2 (1981), 127–132.

[69] Borsik, J., and Doboš, J. On a product of metric spaces. *Mathematica Slovaca 31* (1981), 193–205.

[70] Bortot, S., and Marques Pereira, R. On a new poverty measure constructed from the exponential mean. In *Proc. IFSA/EUSFLAT'15* (Gijón, Spain, 2015), pp. 333–340.

[71] Bortot, S., and Marques Pereira, R. A. The binomial Gini inequality indices and the binomial decomposition of welfare functions. *Fuzzy Sets and Systems 255* (2014), 92–114.

[72] Bottema, O. Het begrip "merkwaardig" met betrekking tot punten in de driehoeksmeetkunde. *Nieuw Tijdschr. Wisk. 69* (1981), 2–7.

[73] Boucher, C., and Ma, B. Closest string with outliers. *BMC Bioinformatics 12* (2011), S55.

[74] Boussou, D., and Perny, P. Ranking methods for valued preference relations: A characterization of a method based on leaving and entering flows. *European Journal of Operational Research 61*, 1–2 (1992), 186–194.

[75] Bouyssou, D., and Marchant, T. Ranking scientists and departments in a consistent manner. *Journal of the American Society for Information Science and Technology 62*, 9 (2011), 1761–1769.

[76] Boyd, S., and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2009.

[77] Boytsov, L. Indexing methods for approximate dictionary searching: Comparative analyses. *ACM Journal of Experimental Algorithmics 16* (2011), 1–86.

[78] Breiman, L. Random forests. *Machine Learning 45* (2001), 5–32.

[79] Bremner, D., Chen, D., Iacono, J., Langerman, S., and Morin, P. Output-sensitive algorithms for Tukey depth and related problems. *Statistics and Computing 18*, 3 (2008), 259–266.

[80] Brent, R. *Algorithms for minimization without derivatives*. Prentice-Hall, 1973.

[81] Brimberg, J. The Fermat-Weber location problem revisited. *Mathematical Programming 71* (1995), 71–76.

[82] Brin, S. Near neighbor search in large metric spaces. In *Proc. Intl. Conf. Very Large Data Bases*. Morgan Kaufmann, 1995, pp. 574–584.

[83] Brönnimann, H., Melquiond, G., and Pionc, S. The design of the Boost interval arithmetic library. *Theoretical Computer Science 351*, 1 (2006), 111–118.

[84] Bronselaer, A., and De Tré, G. Aspects of object merging. In *Proc. NAFIPS'10*. IEEE, Toronto, ON, 2010, pp. 1–6.

[85] Bronselaer, A., Szymczak, M., Zadrożny, S., and De Tré, G. Dynamical order construction in data fusion. *Information Fusion 27* (2016), 1–18.

[86] Brown, B. Statistical uses of the spatial median. *Journal of the Royal Statistical Society. Series B (Methodological) 45*, 1 (1983), 25–30.

[87] Bullen, P. *Handbook of means and their inequalities.* Springer Science+Business Media, Dordrecht, 2003.

[88] Bunke, H. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters 18*, 8 (1997), 689–694.

[89] Bunke, H., and Riesen, K. Recent advances in graph-based pattern recognition with applications in document analysis. *Pattern Recognition 44*, 5 (2011), 1057–1067.

[90] Bunke, H., and Shearer, K. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters 19*, 3–4 (1998), 255–259.

[91] Bustince, H., Barrenechea, E., Calvo, T., James, S., and Beliakov, G. Consensus in multi-expert decision making problems using penalty functions defined over a cartesian product of lattices. *Information Fusion 17* (2014), 56–64.

[92] Bustince, H., Barrenechea, E., and Pagola, M. Relationship between restricted dissimilarity functions, restricted equivalence functions and normal e_N-functions: Image thresholding invariant. *Pattern Recognition Letters 29*, 4 (2008), 525–536.

[93] Bustince, H., Fernandez, J., Kolesárová, A., and Mesiar, R. Fusion functions and directional monotonicity. *Communications in Computer and Information Science 444* (2014), 262–268.

[94] Bustince, H., Fernandez, J., Kolesárová, A., and Mesiar, R. Directional monotonicity of fusion functions. *European Journal of Operational Research 244*, 1 (2015), 300–308.

[95] Bustince, H., Fernandez, J., Mesiar, R., Pradera, A., and Beliakov, G. Restricted dissimilarity functions and penalty functions. In *Proc. Eusflat/LFA 2011* (2011), S. Galichet et al., Eds., pp. 79–85.

[96] Calvo, T., and Beliakov, G. Aggregation functions based on penalties. *Fuzzy Sets and Systems 161* (2010), 1420–1436.

[97] Calvo, T., Kolesárová, A., Komorníková, M., and Mesiar, R. Aggregation operators: Properties, classes and construction methods. In *Aggregation Operators. New Trends and Applications*, T. Calvo, G. Mayor, and R. Mesiar, Eds., vol. 97 of *Studies in Fuzziness and Soft Computing.* Physica-Verlag, New York, 2002, pp. 3–104.

[98] Calvo, T., and Mayor, G. Remarks on two types of extended aggregation functions. *Tatra Mountains Mathematical Publications 16* (1999), 235–253.

[99] Calvo, T., Mesiar, R., and Yager, R. R. Quantitative weights and aggregation. *IEEE Transactions on Fuzzy Systems 12*, 1 (2004), 62–69.

[100] Camargo, J. Must dominance increase with the number of subordinate species in competitive interactions? *Journal of Theoretical Biology 161*, 4 (1993), 537–542.

[101] Carbonell, M., Mas, M., and Mayor, G. On a class of monotonic extended owa operators. In *Proc. 6th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'97)* (Barcelona, Spain, 1997), vol. 3, IEEE, pp. 1695–1700.

[102] Carlsson, C., Fullér, R., and Majlender, P. Additions of completely correlated fuzzy numbers. In *Proc. FUZZ-IEEE'04* (Budapest, Hungary, 2004), IEEE, pp. 535–539.

[103] Carlsson, C., and Fullér, R. On possibilistic mean value and variance of fuzzy numbers. *Fuzzy Sets and Systems 122* (2001), 315–326.

[104] Castagnoli, G., Bräuer, S., and Herrmann, M. Optimization of cyclic redundancy-check codes with 24 and 32 parity bits. *IEEE Transactions on Communications 41*, 6 (1993), 883–892.

[105] Cena, A., and Gagolewski, M. Aggregation and soft clustering of informetric data. In *Proc. 8th International Summer School on Aggregation Operators (AGOP 2015)* (Katowice, Poland, 2015), M. Baczynski, B. De Baets, and R. Mesiar, Eds., University of Silesia, pp. 79–84.

[106] Cena, A., and Gagolewski, M. A K-means-like algorithm for informetric data clustering. In *Proc. IFSA/Eusflat 2015* (2015), J. Alonso, H. Bustince, and M. Reformat, Eds., Atlantic Press, pp. 536–543.

[107] Cena, A., and Gagolewski, M. OM3: Ordered maxitive, minitive, and modular aggregation operators – Axiomatic and probabilistic properties in an arity-monotonic setting. *Fuzzy Sets and Systems 264* (2015), 138–159.

[108] Cena, A., Gagolewski, M., and Mesiar, R. Problems and challenges of information resources producers' clustering. *Journal of Informetrics 9*, 2 (2015).

[109] Chakraborty, B., and Chaudhuri, P. On a transformation and retransformation technique for constructing an affine equivariant multivariate median. *Proceedings of the American Mathematical Society 124*, 8 (1996), 2539–2547.

[110] Chan, T. M. Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete and Computational Geometry 16* (1996), 361–368.

[111] Chan, T. M. An optimal randomized algorithm for maximum Tukey depth. In *Proc. 15th ACM-SIAM Symp. Discrete Algorithms (SODA)* (2004), pp. 430–436.

[112] Chanas, S. On the interval approximation of a fuzzy number. *Fuzzy Sets and Systems 122* (2001), 353–356.

[113] Chaudhuri, P., and Sengupta, D. Sign tests in multidimension: Inference based on the geometry of the data cloud. *Journal of the American Statistical Association 88*, 424 (1993), 1363–1370.

[114] Chavent, M., and Saracco, J. Central tendency and dispersion measures for intervals and hypercubes. *Communications in Statistics – Theory and methods 37* (2008), 1471–1482.

[115] Chazelle, B. An optimal convex hull algorithm in any fixed dimension. *Discrete and Computational Geometry 10*, 1 (1993), 377–409.

[116] Chen, C.-T. Extensions of the TOPSIS for group decision-making under fuzzy environment. *Fuzzy Sets and Systems 114*, 1 (2000), 1–9.

[117] Chen, Z.-Z., and Wang, L. Fast exact algorithms for the closest string and substring problems with application to the planted (l, d) motif model. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*

8, 5 (2011), 1400–1410.

[118] Cheng, Y., and Liu, N. C. A first approach to the classification of the top 500 world universities by their disciplinary characteristics using scientometrics. *Scientometrics 68*, 1 (2006), 135–150.

[119] Chenouri, S., and Small, C. G. A nonparametric multivariate multisample test based on data depth. *Electronic Journal of Statistics 6* (2012), 760–782.

[120] Chimani, M., Woste, M., and Böcker, S. A closer look at the closest string and closest substring problem. In *Proc. 13th Workshop Algorithm Engineering and Experiments (ALENEX'2011)* (2011), pp. 13–24.

[121] Chin, F. Y., Deng, X., Fang, Q., and Zhu, S. Approximate and dynamic rank aggregation. *Theoretical Computer Science 325*, 3 (2004), 409–424.

[122] Choquet, G. Theory of capacities. *Annales de l'institut Fourier 5* (1954), 131–295.

[123] Colomer, J. M., Ed. *Handbook of Electoral System Choice*. Palgrave Macmillan, London, 2004.

[124] Contini, S., Steinfeld, R., Pieprzyk, J., , and Matusiewicz, K. A critical look at cryptographic hash function literature. In *ECRYPT Hash Workshop, 2007* (2007).

[125] Conway, J. H., and Sloane, N. J. A. *Sphere Packings, Lattices and Groups*. Springer-Verlag, New York, 1998.

[126] Coroianu, L. Necessary and sufficient conditions for the equality of the interactive and non-interactive sums of two fuzzy numbers. *Fuzzy Sets and Systems 283* (2016), 40–55.

[127] Coroianu, L., and Fullér, R. On multiplication of interactive fuzzy numbers. In *Proc. IEEE Intl. Symp. Intelligent Systems and Informatics (SISY'13)* (2013), pp. 181–185.

[128] Coroianu, L., Gagolewski, M., and Grzegorzewski, P. Nearest piecewise linear approximation of fuzzy numbers. *Fuzzy Sets and Systems 233* (2013), 26–51.

[129] Coroianu, L., Gagolewski, M., and Grzegorzewski, P. Piecewise linear approximation of fuzzy numbers – a discussion on algorithms, arithmetic operations and stability of fuzzy number characteristics, 2014. Submitted paper.

[130] Costas, R., van Leeuwen, T., and Bordons, M. A bibliometric classificatory approach for the study and assessment of research performance at the individual level: The effects of age on productivity and impact. *Journal of the American Society for Information Science and Technology 61* (2010), 1564–1581.

[131] Couceiro, M., and Marichal, J.-L. Characterizations of discrete Sugeno integrals as polynomial functions over distributive lattices. *Fuzzy Sets and Systems 161* (2010), 694–707.

[132] Cramér, H. *Mathematical methods of statistics*. Princeton University Press, Princeton, 1946.

[133] Czogała, E., and Drewniak, J. Associative monotonic operations in

fuzzy set theory. *Fuzzy Sets and Systems 12* (1984), 249–269.

[134] Damerau, F. J. A technique for computer detection and correction of spelling errors. *Communications of the ACM 7*, 3 (1964), 171–176.

[135] David, H. A., and Nagaraja, H. N. *Order statistics*. Wiley, 2003.

[136] Davis, M., Whistler, K., and Scherer, M. Unicode Technical Standard #10, Unicode Collation Algorithm (revision 30), 2014. http://www.unicode.org/reports/tr10/tr10-30.html.

[137] De Baets, B. Aggregation 2.0. Plenary lecture slides, 7th International Summer School on Aggregation Operators (AGOP'13), Pamplona, Spain, July 16, 2013.

[138] De Baets, B., and Mesiar, R. Triangular norms on product lattices. *Fuzzy Sets and Systems 104* (1999), 61–75.

[139] De Cooman, G., and Kerre, E. Order norms on bounded partially ordered sets. *Journal of Fuzzy Mathematics 2* (1994), 281–310.

[140] de la Rosa de Sáa, S., Gil, M. A., González-Rodríguez, G., López, M. T., and Lubiano, M. A. Fuzzy rating scale-based questionnaires and their statistical analysis. *IEEE Transactions on Fuzzy Systems 23*, 1 (2015), 111–126.

[141] Dean, J., and Ghemawat, S. Mapreduce: Simplified data processing on large clusters. In *Proc. Operating System Design and Implementation (OSDI)* (San Francisco, CA, 2004), pp. 137–150.

[142] del Amo, A., Montero, J., and Molina, E. Representation of recursive rules. *European Journal of Operational Research 130* (2001), 29–53.

[143] Delgado, M., Verdegay, J., and Vila, M. On aggregation operations of linguistic labels. *International Journal of Intelligent Systems 8*, 3 (1993), 351–370.

[144] Delgado, M., Vila, M., and Voxman, W. On a canonical representation of a fuzzy number. *Fuzzy Sets and Systems 93* (1998), 125–135.

[145] Demirci, M. Aggregation operators on partially ordered sets and their categorical foundations. *Kybernetika 42* (2006), 261–277.

[146] Deschrijver, G. Quasi-arithmetic means and OWA functions in interval-valued and Atanassov's intuitionistic fuzzy set theory. In *Proc. Eusflat/LFA 2011* (2011), S. Galichet et al., Eds., pp. 506–513.

[147] Deschrijver, G., and Kerre, E. E. On the relationship between some extensions of fuzzy set theory. *Fuzzy Sets and Systems 133*, 2 (2003), 227–235.

[148] Desu, M. M., and Rodine, R. H. Estimation of the population median. *Skandinavisk Aktuarietidskrift 28* (1969), 67–70.

[149] Diaconis, P., and Graham, R. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society, Series B (Methodological) 39*, 2 (1977), 262–268.

[150] Diaconis, P., and Shahshahani, M. The subgroup algorithm for generating uniform random variables. *Probability In Engineering And Information Sciences 1* (1987), 15–32.

[151] Diamond, P., and Kloeden, P. *Metric spaces of fuzzy sets. Theory and applications*. World Scientific, Singapore, 1994.

[152] Didehvar, F., and Eslahchi, C. An algorithm for rank aggregation problem. *Applied Mathematics and Computation 189*, 2 (2007), 1847–1858.

[153] Dinu, L. P. On the classification and aggregation of hierarchies with different constitutive elements. *Fundamenta Informaticæ 55*, 1 (2003), 39–50.

[154] Dinu, L. P., and Ionescu, R.-T. Clustering methods based on closest string via rank distance. In *14th Intl. Symp. Symbolic and Numeric Algorithms for Scientific Computing* (2012), IEEE, pp. 207–213.

[155] Dinu, L. P., and Ionescu, R.-T. An efficient rank based approach for closest string and closest substring. *PLoS One 7*, 6 (2012), e37576.

[156] Dinu, L. P., and Manea, F. An efficient approach for the rank aggregation problem. *Theoretical Computer Science 359*, 1–3 (2006), 455–461.

[157] Dinu, L. P., and Popa, A. On the closest string via rank distance. *Lecture Notes in Computer Science 7354* (2012), 413–426.

[158] Domingo-Ferrer, J., and Torra, V. Disclosure risk assessment in statistical microdata protection via advanced record linkage. *Statistics and Computing 13* (2003), 343–354.

[159] Donoho, D. *Breakdown properties of multivariate location estimates.* PhD thesis, Department of Statistics, Harvard University, 1982.

[160] Donoho, D. L., and Gasko, M. Breakdown properties of location estimates based on halfspace depth and projected outlyingness. *The Annals of Statistics 20*, 4 (1992), 1803–1827.

[161] Dubois, D., Fargier, H., and Prade, H. Refinements of the maximin approach to decision-making in a fuzzy environment. *Fuzzy Sets and Systems 81* (1996), 103–122.

[162] Dubois, D., Fortemps, P., Pirlot, M., and Prade, H. Leximin optimality and fuzzy set theoretic operations. *European Journal of Operational Research 130*, 1 (2001), 20–28.

[163] Dubois, D., Kerre, E., Mesiar, R., and Prade, H. Fuzzy interval analysis. In *Fundamentals of fuzzy sets*, D. Dubois and H. Prade, Eds. Kluwer, Boston, Mass., 2000, pp. 483–581.

[164] Dubois, D., and Prade, H. Operations on fuzzy numbers. *Int. J. Syst. Sci. 9* (1978), 613–626.

[165] Dubois, D., and Prade, H. *Fuzzy sets and systems. Theory and applications.* Academic Press, New York, 1980.

[166] Dubois, D., and Prade, H. A review of fuzzy set aggregation connectives. *Information Sciences 39* (1985), 85–121.

[167] Dubois, D., and Prade, H. The mean value of a fuzzy number. *Fuzzy Sets and Systems 24* (1987), 279–300.

[168] Dubois, D., and Prade, H. Semantics of quotient operators in fuzzy relational databases. *Fuzzy Sets and Systems 78*, 1 (1996), 89–93.

[169] Dubois, D., and Prade, H. On the use of aggregation operations in information fusion processes. *Fuzzy Sets and Systems 142* (2004), 143–161.

[170] Dubois, D., Prade, H., and Testemale, C. Weighted fuzzy pattern matching. *Fuzzy Sets and Systems 28* (1988), 313–331.

[171] Dujmović, J. J. Two integrals related to means. *Publikacije Elektrotehničkog Fakulteta Univerziteta u Beogradu 412–460*, 457 (1974), 231–232.

[172] Dujmović, J. J. Weighted conjunctive and disjunctive means and their application in system evaluation. *Publikacije Elektrotehničkog Fakulteta Univerziteta u Beogradu 461–497*, 483 (1974), 147–158.

[173] Dukhovny, A. Lattice polynomials of random variables. *Statistics and Probability Letters 77* (2007), 989–994.

[174] Durocher, S., Fraser, R., Leblanc, A., Morrison, J., and Skala, M. On combinatorial depth measures. In *Proc. 26th Canadian Conf. Computational Geometry* (2014), pp. 206–211.

[175] Dyckerhoff, R., Koshevoy, G., and Mosler, K. Zonoid data depth: Theory and computation. In *Proc. COMPSTAT 1996* (Heidelberg, 1996), A. Prat et al., Eds., Physica-Verlag, pp. 235–240.

[176] Eaton, M. L. *Multivariate Statistics.* Wiley, New York, 1983.

[177] Eddelbuettel, D. *Seamless* R *and* C++ *Integration with* Rcpp. Springer, New York, 2013.

[178] Eddy, W. Convex hull peeling. In *Proc. COMPSTAT'82* (Vienna, 1982), Physica-Verlag, pp. 42–47.

[179] Edelsbrunner, H. *Algorithms in Combinatorial Geometry.* Springer-Verlag, Heidelberg, 1987.

[180] Egghe, L. An improvement of the h-index: the g-index. *ISSI Newsletter 2*, 1 (2006), 8–9.

[181] Ehrenfeucht, A., and Haussler, D. A new distance metric on strings computable in linear time. *Discrete Applied Mathematics 20* (1988), 191–203.

[182] Even, Y., and Lehrer, E. Decomposition-integral: Unifying Choquet and the concave integrals. *Economic Theory 56*, 1 (2014), 33–58.

[183] Fan, K. Entfernung zweier zufälligen Größen und die Konvergenz nach Wahrscheinlichkeit. *Mathematische Zeitschrift 49* (1943), 681–683.

[184] Fernández, M.-L., and Valiente, G. A graph distance metric combining maximum common subgraph and minimum common supergraph. *Pattern Recognition Letters 22*, 6–7 (2001), 753–758.

[185] Fernández Salido, J., and Murakami, S. Extending Yager's orness concept for the OWA aggregators to other mean operators. *Fuzzy Sets and Systems 139*, 3 (2003), 515–542.

[186] Filev, D., and Yager, R. R. On the issue of obtaining OWA operator weights. *Fuzzy Sets and Systems 94* (1998), 157–169.

[187] Fischer, K., Gärtner, B., and Kutz, M. Fast smallest-enclosing-ball computation in high dimensions. In *Proc. 11th European Symposium on Algorithms (ESA)* (2003), pp. 630–641.

[188] Fishburn, P. Condorcet social choice functions. *SIAM Journal on Applied Mathematics 33*, 3 (1977), 469–489.

[189] Fishburn, P. C. Lexicographic orders, utilities and decision rules: A survey. *Management Science 20*, 11 (1974), 1442–1471.

[190] Fisher, N. *Statistical Analysis of Circular Data.* Cambridge University

Press, 1993.

[191] Fisher, R. The correlation between relatives on the supposition of Mendelian inheritance. *Philosophical Transactions of the Royal Society of Edinburgh 52* (1918), 399–433.

[192] Fisher, R. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society A 222* (1922), 309–368.

[193] Fisher, R. A., and Yates, F. *Statistical tables for biological, agricultural and medical research*. Oliver & Boyd, London, 1938.

[194] Floyd, R., and Rivest, R. Expected time bounds for selection. *Communications of the ACM 18*, 3 (1975), 165–172.

[195] Fodor, J. An extension of Fung-Fu's theorem. *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems 4*, 3 (1996), 235–243.

[196] Fodor, J., Marichal, J.-L., and Roubens, M. Characterization of the Ordered Weighted Averaging operators. *IEEE Transactions on Fuzzy Systems 3*, 2 (1995), 236–240.

[197] Forgy, E. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics 21* (1965), 768–769.

[198] Frances, M., and Litman, A. On covering problems of codes. *Theory of Computing Systems 30*, 2 (1997), 113–119.

[199] Franceschini, F., and Maisano, D. A. The Hirsch index in manufacturing and quality engineering. *Quality and Reliability Engineering International 25* (2009), 987–995.

[200] Franceschini, F., and Maisano, D. A. Structured evaluation of the scientific output of academic research groups by recent h-based indicators. *Journal of Informetrics 5* (2011), 64–74.

[201] Fraser, A. Simulation of genetic systems by automatic digital computers. I. Introduction. *Australian Journal of Biological Sciences 10* (1957), 484–491.

[202] Fraser, A., and Burnell, D. *Computer Models in Genetics*. McGraw-Hill, New York, 1970.

[203] Fullér, R., and Majlender, P. On interactive fuzzy numbers. *Fuzzy Sets and Systems 143* (2003), 355–369.

[204] Gagolewski, M. On the relation between effort-dominating and symmetric minitive aggregation operators. In *Advances in Computational Intelligence, Part III*, S. Greco et al., Eds., vol. 299. Springer, 2012, pp. 276–285.

[205] Gagolewski, M. On the relationship between symmetric maxitive, minitive, and modular aggregation operators. *Information Sciences 221* (2013), 170–180.

[206] Gagolewski, M. Statistical hypothesis test for the difference between Hirsch indices of two Pareto-distributed random samples. In *Synergies of Soft Computing and Statistics for Intelligent Data Analysis*, R. Kruse et al., Eds., vol. 190. Springer, 2013, pp. 359–367.

[207] Gagolewski, M. Normalized WD_pWAM and WD_pOWA spread measures. In *Proc. IFSA/Eusflat 2015* (2015), J. Alonso, H. Bustince, and M. Reformat, Eds., Atlantic Press, pp. 210–216.

[208] Gagolewski, M. Some issues in aggregation of multidimensional data.

In *Proc. 8th International Summer School on Aggregation Operators (AGOP 2015)* (Katowice, Poland, 2015), M. Baczynski, B. De Baets, and R. Mesiar, Eds., University of Silesia, pp. 127–132.

[209] Gagolewski, M. Spread measures and their relation to aggregation functions. *European Journal of Operational Research 241*, 2 (2015), 469–477.

[210] Gagolewski, M., and Caha, J. `FuzzyNumbers`*: Tools to deal with fuzzy numbers in R*, 2015. http://FuzzyNumbers.rexamine.com, doi:10.5281/zenodo.15677.

[211] Gagolewski, M., and Grzegorzewski, P. A geometric approach to the construction of scientific impact indices. *Scientometrics 81*, 3 (2009), 617–634.

[212] Gagolewski, M., and Grzegorzewski, P. Arity-monotonic extended aggregation operators. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, E. Hüllermeier et al., Eds., vol. 80. Springer, 2010, pp. 693–702.

[213] Gagolewski, M., and Grzegorzewski, P. S-statistics and their basic properties. In *Combining Soft Computing and Statistical Methods in Data Analysis*, C. Borgelt et al., Eds. Springer, 2010, pp. 281–288.

[214] Gagolewski, M., and Grzegorzewski, P. Possibilistic analysis of arity-monotonic aggregation operators and its relation to bibliometric impact assessment of individuals. *International Journal of Approximate Reasoning 52*, 9 (2011), 1312–1324.

[215] Gagolewski, M., and Mesiar, R. Aggregating different paper quality measures with a generalized h-index. *Journal of Informetrics 6*, 4 (2012), 566–579.

[216] Gagolewski, M., and Mesiar, R. Monotone measures and universal integrals in a uniform framework for the scientific impact assessment problem. *Information Sciences 263* (2014), 166–174.

[217] Gao, X., Xiao, B., Tao, D., and Li, X. A survey of graph edit distance. *Pattern Analysis and Applications 13*, 1 (2010), 113–129.

[218] García-Lapresta, J., Lasso de la Vega, C., Marques Pereira, R., and Urrutia, A. A new class of fuzzy poverty measures. In *Proc. of IFSA/EUSFLAT2015* (Gijón, Spain, 2015), pp. 1140–1146.

[219] Gärtner, B. Fast and robust smallest enclosing balls. *Lecture Notes in Computer Science 1643* (1999), 325–338.

[220] Gärtner, B., and Schönherr, S. An efficient, exact, and generic quadratic programming solver for geometric optimization. In *Proc. 16th ACM Symposium on Computational Geometry* (2000), pp. 110–118.

[221] Genz, A., and Malik, A. An adaptive algorithm for numeric integration over an n-dimensional rectangular region. *Journal of Computational and Applied Mathematics 6*, 4 (1980), 295–302.

[222] Ghiselli Ricci, R. Finitely and absolutely non idempotent aggregation operators. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 12*, 2 (2004), 201–217.

[223] Ghiselli Ricci, R. Asymptotically idempotent aggregation operators. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Sys-*

tems 17, 5 (2009), 611–631.

[224] Gionis, A., Mannila, H., and Tsaparas, P. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data 1*, 1 (2007), 4.

[225] Godo, L., and Torra, V. On aggregation operators for ordinal qualitative information. *IEEE Transactions on Fuzzy Systems 8*, 2 (2000), 143–154.

[226] Goldberg, D. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys 21*, 1 (1991), 5–48.

[227] Goldfarb, D., and Idnani, A. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming 27* (1983), 1–33.

[228] Gower, J. C. Algorithm AS 78: The Mediancentre. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 23*, 3 (1974), 466–470.

[229] Grabisch, M. k-order additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems 92* (1997), 167–189.

[230] Grabisch, M., Marichal, J.-L., Mesiar, R., and Pap, E. *Aggregation functions*. Cambridge University Press, 2009.

[231] Gramm, J., Niedermeier, R., and Rossmanith, P. Fixed-parameter algorithms for closest string and related problems. *Algorithmica 37* (2003), 25–42.

[232] Greco, S., Mesiar, R., and Rindone, F. Two new characterizations of universal integrals on the scale $[0, 1]$. *Information Sciences 267* (2014), 217–224.

[233] Green, P. Peeling bivariate data. In *Interpreting multivariate data*, V. Barnett, Ed. Wiley, New York, 1981.

[234] Grübel, R. Orthogonalization of multivariate location estimators: The orthomedian. *The Annals of Statistics 24*, 4 (1996), 1457–1473.

[235] Grzegorzewski, P. Metrics and orders in space of fuzzy numbers. *Fuzzy Sets and Systems 97* (1998), 83–94.

[236] Grzegorzewski, P. Distances between intuitionistic fuzzy sets and/or interval-valued fuzzy sets based on the Hausdorff metric. *Fuzzy Sets and Systems 148*, 2 (2004), 319–328.

[237] Grzegorzewski, P. Granular regression. In *Proc. IFSA/NAFIPS'13* (Edmonton, Canada, 2013), pp. 974–979.

[238] Halmos, P. *Measure Theory*. Van Nostrand, New York, 1950.

[239] Hamming, R. W. Error detecting and error correcting codes. *Bell System Technical Journal 29*, 2 (1950), 147–160.

[240] Hansen, N. The CMA evolution strategy: A comparing review. In *Towards a new evolutionary computation. Advances in estimation of distribution algorithms* (2006), J. Lozano, P. Larranga, I. Inza, and E. Bengoetxea, Eds., Springer, pp. 75–102.

[241] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer, 2013.

[242] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, 2013.

[243] He, X., and Shi, P. Monotone b-spline smoothing. *Journal of the American Statistical Association 93*, 442 (1998).

[244] Heip, C. A new index measuring evenness. *Journal of Marine Biological Association of the United Kingdom 54*, 3 (1974), 555–557.

[245] Herrera, F., Herrera-Viedma, E., and Verdegay, J. Direct approach processes in group decision making using linguistic OWA operators. *Fuzzy Sets and Systems 79*, 2 (1996), 175–190.

[246] Higham, N. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, 2002.

[247] Higham, N. J. The accuracy of floating point summation. *SIAM Journal on Scientific Computing 14*, 4 (1993), 783–799.

[248] Hilbert, D. Über die stetige Abbildung einer Linie auf ein Flächenstück. *Mathematische Annalen 38* (1891), 459–460.

[249] Hirsch, J. E. An index to quantify individual's scientific research output. *Proceedings of the National Academy of Sciences 102*, 46 (2005), 16569–16572.

[250] Hoare, C. Algorithm 65: Find. *Communications of the ACM 4*, 7 (1961), 321–322.

[251] Hryniewicz, O. Statistics with fuzzy data in statistical quality control. *Soft Computing 12*, 3 (2008), 229–234.

[252] Huber, P. J. The 1972 wald lecture robust statistics: A review. *Annals of Mathematical Statistics 42*, 4 (1972), 1041–1067.

[253] Huber, P. J. Projection pursuit. *The Annals of Statistics 13*, 2 (1985), 435–475.

[254] Hufsky, F., Kuchenbecker, L., Jahn, K., Stoye, J., and Böcker, S. Swiftly computing center strings. *BMC Bioinformatics 12* (2011), 106.

[255] Hyndman, R. J., and Fan, Y. Sample quantiles in statistical packages. *The American Statistician 50*, 4 (1996), 361–365.

[256] Ibàñez, A., Larrañaga, P., and Bielza, C. Cluster methods for assessing research performance: Exploring Spanish computer science. *Scientometrics 97*, 3 (2013), 571–600.

[257] Irpino, A., and Verde, R. Dynamic clustering of interval data using a wasserstein-based distance. *Pattern Recognition Letters 29*, 11 (2008), 1648–1658.

[258] Jamison, B., Orey, S., and Pruitt, W. Convergence of weighted averages of independent random variables. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete 4*, 1 (1965), 40–44.

[259] Jammalamadaka, S. R., and SenGupta, A. *Topics in Circular Statistics*. World Scientific Press, Singapore, 2001.

[260] Jaroszewicz, S., and Korzeń, M. Arithmetic operations on independent random variables: A numerical approach. *SIAM Journal on Scientific Computing 34* (2012), A1241–A1265.

[261] Jenei, S., and De Baets, B. On the direct decomposability of t-norms on product lattices. *Fuzzy Sets and Systems 139*, 3 (2003), 699–707.

[262] Jiang, X., Wentker, J., and Ferrer, M. Generalized median string computation by means of string embedding in vector spaces. *Pattern*

Recognition Letters 33 (2012), 842–852.

[263] Johnson, R. *Modern Geometry: An Elementary Treatise on the Geometry of the Triangle and the Circle*. Houghton Mifflin, Boston, MA, 1929.

[264] Juan, A., and Vidal, E. Fast median search in metric spaces. *Lecture Notes in Computer Science 1451* (1998), 905–912.

[265] Kacprzyk, J., and Zadrożny, S. Computing with words for text categorization. *Studies in Fuzziness and Soft Computing 209* (2007), 339–362.

[266] Kahan, W. Further remarks on reducing truncation errors. *Communications of the ACM 8*, 1 (1965), 40.

[267] Karaçal, F., and Mesiar, R. Uninorms on bounded lattices. *Fuzzy Sets and Systems 261* (2015), 33–43.

[268] Kärkkäinen, T., and Äyrämö, S. On computation of spatial median for robust data mining. In *Proc. EUROGEN 2005* (2005), R. Schilling et al., Eds., pp. 1–14.

[269] Kemeny, J. G. Mathematics without numbers. *Daedalus 88*, 4 (1959), 577–591.

[270] Kerre, E. E. A tribute to Zadeh's extension principle. *Scientia Iranica 18*, 3 (2011), 593–595.

[271] Kim, W. J., Ko, J. H., and Chung, M. J. Uncertain robot environment modelling using fuzzy numbers. *Fuzzy Sets and Systems 61*, 1 (1994), 53–62.

[272] Kimberling, C. Central points and central lines in the plane of a triangle. *Mathematics Magazine 67*, 3 (1994), 163–187.

[273] Kimberling, C. Triangle centers and central triangles. *Congressus Numerantium 129* (1998), 1–295.

[274] Kitagawa, T. On some class of weighted means. *Proceedings of the Physico-Mathematical Society of Japan 16* (1934).

[275] Kleene, S. C. On the forms of the predicates in the theory of constructive ordinals. *American Journal of Mathematics 77*, 3 (1955), 405–428.

[276] Klement, E., Mesiar, R., and Pap, E. A universal integral as common frame for Choquet and Sugeno integral. *IEEE Transactions on Fuzzy Systems 18* (2010), 178–187.

[277] Klement, E. P., Mesiar, R., and Pap, E. *Triangular norms*. Kluwer Academic Publishers, 2000.

[278] Klement, E. P., Mesiar, R., and Pap, E. Triangular norms. Position paper I: Basic analytical and algebraic properties. *Fuzzy Sets and Systems 143* (2004), 5–26.

[279] Klement, E. P., Mesiar, R., and Pap, E. Triangular norms. Position paper II: General constructions and parametrized families. *Fuzzy Sets and Systems 145* (2004), 411–438.

[280] Klement, E. P., Mesiar, R., and Pap, E. Triangular norms. Position paper III: Continuous t-norms. *Fuzzy Sets and Systems 145* (2004), 439–454.

[281] Klir, G. J., and Yuan, B. *Fuzzy sets and fuzzy logic. Theory and applications*. Prentice Hall PTR, New Jersey, 1995.

[282] Knuth, D. E. *The Art of Computer Programming. Volume 2. Seminu-*

merical Algorithms. Addison Wesley, Reading, MA, 1998.

[283] Kobus, M. Attribute decomposition of multidimensional inequality indices. *Economics Letters 117*, 1 (2012), 189–191.

[284] Kobus, M., and Miłoś, P. Inequality decomposition by population subgroups for ordinal data. *Journal of Health Economics 31*, 1 (2012), 15–21.

[285] Kohonen, T., and Somervuo, P. J. Self-organizing maps of symbol strings. *Neurocomputing 21* (1998), 19–30.

[286] Kojadinovic, I. Unsupervised aggregation by the choquet integral based on entropy functionals: Application to the evaluation of students. *Lecture Notes in Computer Science 3131* (2004), 163–174.

[287] Kojadinovic, I. Unsupervised aggregation of commensurate correlated attributes by means of the choquet integral and entropy functionals. *International Journal of Intelligent Systems 23*, 2 (2008), 128–154.

[288] Kojadinovic, I., and Marichal, J.-L. On the moments and distribution of discrete choquet integrals from continuous distributions. *Journal of Computational and Applied Mathematics 230* (2009), 83–94.

[289] Kołacz, A., and Grzegorzewski, P. Measures of dispersion for multidimensional data, 2015. Submitted paper.

[290] Kolesárová, A., Mayor, G., and Mesiar, R. Weighted ordinal means. *Information Sciences 177* (2007), 3822–3830.

[291] Kolesárová, A., Mesiar, R., and Montero, J. Sequential aggregation of bags. *Information Sciences 294* (2015), 305–314.

[292] Kolmogorov, A. Sur la notion de la moyenne. *Atti della R. Academia nazionale dei Lincei 12* (1930), 388–391.

[293] Komorníková, M., and Mesiar, R. Aggregation functions on bounded partially ordered sets and their classification. *Fuzzy Sets and Systems 175* (2011), 48–56.

[294] Konohen, T. Median strings. *Pattern Recognition Letters 3* (1985), 309–313.

[295] Korzeń, M., and Jaroszewicz, S. PaCAL: A Python package for arithmetic computations with random variables. *Journal of Statistical Software 57*, 10 (2014), 1–34.

[296] Koshevoy, G., and Mosler, K. Zonoid trimming for multivariate distributions. *The Annals of Statistics 25*, 5 (1997).

[297] Kosmulski, M. A new Hirsch-type index saves time and works equally well as the original *h*-index. *ISSI Newsletter 2*, 3 (2006), 4–6.

[298] Kosmulski, M. MAXPROD — A new index for assessment of the scientific output of an individual, and a comparison with the *h*-index. *Cybermetrics 11*, 1 (2007), 5.

[299] Kostal, L., Lansky, P., and Pokora, O. Measures of statistical dispersion based on Shannon and Fisher information concepts. *Information Sciences 235* (2013), 214–223.

[300] Krarup, J., and Vajda, S. On Torricelli's geometrical solution to a problem of Fermat. *IMA Journal of Management Mathematics 8* (1997), 215–223.

[301] Kruskal, J. B. An overview of sequence comparison: Time warps, string edits, and macromolecules. *SIAM Review 25*, 2 (1983), 201–237.

[302] Kulczycki, P., and Kowalski, P. A. Bayes classification of imprecise information of interval type. *Control and Cybernetics 40*, 1 (2011), 101–123.

[303] Kullback, S., and Leibler, R. On information and sufficiency. *Annals of Mathematical Statistics 22*, 1 (1951), 79–86.

[304] Kwakernaak, H. Fuzzy random variables: I. Definitions and theorems. *Information Sciences 15*, 1 (1978), 1–29.

[305] Lanctot, J. K., Li, M., Ma, B., Wang, S., and Zhang, L. Distinguishing string selection problems. *Information and Computation 185* (2003), 41–55.

[306] Langerman, S., and Steiger, W. Computing a high depth point in the plane. In *Developments in Robust Statistics* (2003), pp. 228–234.

[307] Lázaro, J., and Calvo, T. XAO operators – The interval universe. In *Proc. Eusflat/LFA 2005* (2005), pp. 189–197.

[308] Le Gall, F. Powers of tensors and fast matrix multiplication. In *Proc. 39th Intl. Symp. Symbolic and Algebraic Computation (ISSAC'14)* (New York, 2014), ACM, pp. 296–303.

[309] Lee, E. A simplified B-spline computation routine. *Computing 29*, 4 (1982), 365–371.

[310] Lehmann, E., and Casella, G. *Theory of Point Estimation*. Springer, New York, 1988.

[311] Lehmann, E. L. Ordered families of distributions. *Annals of Mathematical Statistics 26* (1955), 399–419.

[312] Lehrer, E. A new integral for capacities. *Economic Theory 39*, 1 (2009), 157–176.

[313] Lehtonen, E., Marichal, J.-L., and Teheux, B. Associative string functions. *Asian-European Journal of Mathematics 7* (2014), 1450059.

[314] Leisch, F. A toolbox for K-centroids cluster analysis. *Computational Statistics & Data Analysis 51*, 2 (2006), 526–544.

[315] Lenstra Jr., H. Integer programming with a fixed number of variables. *Mathematics of Operations Research 8*, 4 (1983), 538–548.

[316] Lessmann, M., and Würtz, R. P. Fast nearest neighbor search in pseudosemimetric spaces. In *Proc. VISAPP'12* (2012), pp. 667–674.

[317] Levenshtein, V. I. Binary codes capable of correcting deletions, insertions, or reversals. *Soviet Physics Doklady 10*, 8 (1966), 707–710.

[318] Ley, C., Sabbah, C., and Verdebout, T. A new concept of quantiles for directional data and the angular Mahalanobis depth. *Electronic Journal of Statistics 8*, 1 (2014), 795–816.

[319] Li, J., and Liu, R. Y. New nonparametric tests of multivariate locations and scales using data depth. *Statistical Science 19*, 4 (2004), 686–696.

[320] Li, M., Ma, B., and Wang, L. On the closest string and substring problems. *Journal of the ACM 49*, 2 (2002), 157–171.

[321] Lin, S. Rank aggregation methods. *Wiley Interdisciplinary Reviews: Computational Statistics 2*, 5 (2010), 555–570.

[322] Lipschitz, R. O. S. De explicatione per series trigonometricas instituenda functionum unius variabilis arbitrariarum, et praecipue earum, quae per variabilis spatium finitum valorum maximourm et minimorum numerum habent infinitum, disquisitio. *Journal für die reine und angewandte Mathematik 63*, 2 (1864), 296–308.

[323] Liu, R. Y. On a notion of data depth based on random simplices. *Annals of Statistics 18* (1990), 405–414.

[324] Liu, R. Y., Parelius, J. M., and Singh, K. Multivariate analysis by data depth: Descriptive statistics, graphics and inference. *The Annals of Statistics 27*, 3 (1999), 783–858.

[325] Liu, R. Y., and Singh, K. Ordering directional data: Concepts of data depth on circles and spheres. *The Annals of Statistics 20*, 3 (1992), 1468–1484.

[326] Lizasoain, I., and Moreno, C. OWA operators defined on complete lattices. *Fuzzy Sets and Systems 224* (2013), 36–52.

[327] Lopuhaä, H. P., and Rousseeuw, P. J. Breakdown points of affine equivariant estimators of multivariate location and covariance matrics. *The Annals of Statistics 19*, 1 (1991), 229–248.

[328] Lovisolo, L., and da Silva, E. A. B. Uniform distribution of points on a hyper-sphere with applications to vector bit-plane encoding. *IEE Proceedings on Vision, Image and Signal Processing 148*, 3 (2001), 187–193.

[329] Lowrance, R., and Wagner, R. A. An extension of the string-to-string correction problem. *Journal of the ACM 22*, 2 (1975), 177–183.

[330] Lucca, G., Sanz, J., Pereira Dimuro, G., Bedregal, B., Mesiar, R., Kolesárová, A., and Bustince, H. Pre-aggregation functions: construction and an application. *IEEE Transactions on Fuzzy Systems* (2015). In press, doi:10.1109/TFUZZ.2015.2453020.

[331] MacQueen, J. B. Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.* (Berkeley, 1967), vol. 1, University of California Press, pp. 281–297.

[332] Mardia, K. Statistics of directional data. *Journal of the Royal Statistical Society. Series B (Methodological) 37*, 3 (1975), 349–393.

[333] Mardia, K., and Jupp, E. *Directional Statistics*. Wiley, 1999.

[334] Marichal, J.-L. An axiomatic approach of the discrete choquet integral as a tool to aggregate interacting criteria. *IEEE Transactions on Fuzzy Systems 8*, 6 (2000), 800–807.

[335] Marichal, J.-L. On Sugeno integral as an aggregation function. *Fuzzy Sets and Systems 114* (2000), 347–365.

[336] Marichal, J.-L. On the associativity functional equation. *Fuzzy Sets and Systems 114*, 3 (2000), 381–389.

[337] Marichal, J.-L. On order invariant synthesizing function. *Journal of Mathematical Psychology 46*, 6 (2002), 661–676.

[338] Marichal, J.-L. Cumulative distribution functions and moments of lattice polynomials. *Statistics and Probability Letters 76* (2006), 1273–1279.

[339] Marichal, J.-L. k-intolerant capacities and choquet integrals. *European*

Journal of Operational Research 177, 3 (2007), 1453–1468.

[340] Marichal, J.-L. Weighted lattice polynomials of independent random variables. *Discrete Applied Mathematics 156* (2008), 685–694.

[341] Marichal, J.-L. Weighted lattice polynomials. *Discrete Mathematics 309* (2009), 814–820.

[342] Marichal, J.-L., and Kojadinovic, I. Distribution functions of linear combinations of lattice polynomials from the uniform distribution. *Statistics and Probability Letters 78* (2008), 985–991.

[343] Marichal, J.-L., and Mesiar, R. Aggregation on finite ordinal scales by scale independent functions. *Order 21*, 2 (2004), 155–180.

[344] Marichal, J.-L., and Rubens, M. Characterization of some stable aggregation functions. In *Proc. 1st Conf. Industrial Engineering and Production Management (IEPM'93)* (1993), pp. 187–196.

[345] Marsaglia, G. Choosing a point from the surface of a sphere. *Annals of Mathematical Statistics 43* (1972), 645–646.

[346] Martin, J., and Mayor, G. Some properties of multi-argument distances and Fermat multidistance. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems* (2010), E. Hüllermeier et al., Eds., vol. 80, Springer-Verlag, pp. 703–711.

[347] Martin, J., and Mayor, G. Multi-argument distances. *Fuzzy Sets and Systems 167* (2011), 92–100.

[348] Martín, J., Mayor, G., and Suñer, J. On dispersion measures. *Mathware & Soft Computing 8* (2001), 227–237.

[349] Martin, J., Mayor, G., and Valero, O. A fixed point theorem for asymmetric distances via aggregation functions. In *Proc. 6th Intl. Summer School on Aggregation Operators (AGOP 2011)* (Benevento, Italy, 2011), pp. 217–222.

[350] Martínez-Hinarejos, C., Juan, A., and Casacuberta, F. Median strings for k-nearest neighbour classification. *Pattern Recognition Letters* (2003), 173–181.

[351] Marzal, A., and Vidal, E. Computation of normalized edit distance and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*, 9 (1993), 926–932.

[352] Masek, W. J., and Pateson, M. S. A faster algorithm computing string edit distances. *Journal of Computer and System Sciences 20* (1980), 18–31.

[353] Massé, J.-C. Multivariate trimmed means based on the Tukey depth. *Journal of Statistical Planning and Interference 139* (2009), 366–384.

[354] Massé, J.-C., and Plante, J. F. A Monte Carlo study of the accuracy and robustness of ten bivariate location estimators. *Computational Statistics & Data Analysis 42* (2003), 1–26.

[355] Mayor, G., and Calvo, T. On extended aggregation functions. In *Proc. IFSA 1997* (Prague, 1997), vol. 1, Academia, pp. 281–285.

[356] Mays, E., Damerau, F. J., and Mercer, R. L. Context based spelling correction. *Information Processing & Management 27*, 2 (1991), 517–522.

[357] Mazumdar, A., Polyanskiy, Y., and Saha, B. On Chebyshev radius

of a set in Hamming space and the closest string problem. In *Proc. IEEE Intl. Symp. Information Theory* (2013), IEEE, pp. 1401–1405.

[358] Mendel, F., Nad, T., and Schläffer, M. Improving local collisions: New attacks on reduced SHA-256. *Lecture Notes in Computer Science 7881* (2013), 262–278.

[359] Meneses, C. N., Lu, Z., Oliveira, C. A. S., and Pardalos, P. M. Optimal solutions for the closest-string problem via integer programming. *INFORMS Journal on Computing 16*, 4 (2004), 419–429.

[360] Mesiar, R. Integration based on decomposition. Seminar tutorial slides, Warsaw, Poland, December 11, 2014.

[361] Mesiar, R. Fuzzy set approach to the utility, preference relations, and aggregation operators. *European Journal of Operational Research 176* (2007), 414–422.

[362] Mesiar, R., and Mesiarová-Zemánková, A. The ordered modular averages. *IEEE Transactions on Fuzzy Systems 19*, 1 (2011), 42–50.

[363] Mesiar, R., and Pap, E. Aggregation of infinite sequences. *Information Sciences 178* (2008), 3557–3564.

[364] Mesiar, R., and Stupňanová, A. Decomposition integrals. *International Journal of Approximate Reasoning 54*, 8 (2013), 1252–1259.

[365] Micó, L., and Oncina, J. An approximate median search algorithm in non-metric spaces. *Pattern Recognition Letters 22* (2001), 1145–1151.

[366] Milasevic, P., and Ducharme, G. Uniqueness of the spatial median. *The Annals of Statistics 15*, 3 (1987), 1332–1333.

[367] Miroiu, A. Axiomatizing the hirsch index: Quantity and quality disjoined. *Journal of Informetrics 7* (2013), 10–15.

[368] Morgan, H. L. Spelling correction in systems programs. *Journal of the ACM 13*, 2 (1970), 90–94.

[369] Möttönen, J., Nordhausen, K., and Oja, H. Asymptotic theory of the spatial median. *Nonparametrics and Robustness in Modern Statistical Inference and Time Series 7* (2010), 182–193.

[370] Nagumo, M. Über eine Klasse der Mittelwerte. *Japanese Journal of Mathematics 7* (1930), 71–79.

[371] Navarro, G. A guided tour to approximate string matching. *ACM Computing Surveys 33*, 1 (2001), 31–88.

[372] Needleman, S., and Wunsch, C. D. A general method applicable to the search of similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology 48* (1970), 443–453.

[373] Nelsen, R. *An Introduction to Copulas*. Springer-Verlag, 1999.

[374] Nicolas, F., and Rivals, E. Complexities of the centre and median string problems. *Lecture Notes in Computer Science 2676* (2003), 315–327.

[375] Nicolas, F., and Rivals, E. Hardness results for the center and median string problems under the weighted and unweighted edit distances. *Journal of Discrete Algorithms 3*, 2–4 (2005), 390–415.

[376] Niinimaa, A., Oja, H., and Tableman, M. The finite-sample breakdown point of the oja bivariate median and of the corresponding half-

samples version. *Statistics & Probability Letters 10* (1990), 325–328.

[377] Nocedal, J., and Wright, S. *Numerical Optimization.* Springer-Verlag, New York, 2006.

[378] Oja, H. Descriptive statistics for multivariate distributions. *Statistics & Probability Letters 1* (1983), 327–332.

[379] Oommen, B. Constrained string editing. *Information Sciences 40* (1986), 267–284.

[380] Ortega, J. L., López-Romero, E., and Fernández, I. Multivariate approach to classify research institutes according to their outputs: The case of the CSIC's institutes. *Journal of Informetrics 5* (2011), 323–332.

[381] Otieno, B. S. *An Alternative Estimate of Preferred Direction for Circular Data.* PhD thesis, Virginia Polytechnic Institute and State University, 2002.

[382] Ovchinnikov, S. Means on ordered sets. *Mathematical Social Sciences 32* (1996), 39–56.

[383] Ovchinnikov, S. Invariant functions on simple orders. *Order 14* (1998), 365–371.

[384] Park, H.-S., and Jun, C.-H. A simple and fast algorithm for K-medoids clustering. *Expert Systems with Applications 36* (2009), 3336–3341.

[385] Pearson, K. Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society A 185* (1894), 71–110.

[386] Pedrycz, W., Skowron, A., and Kreinovich, V., Eds. *Handbook of Granular Computing.* John Wiley and Sons, Chichester, 2008.

[387] Peters, G. Granular box regression. *IEEE Transactions on Fuzzy Systems 19* (2011), 1141–1152.

[388] Peters, G., and Lacic, Z. Tackling outliers in granular box regression. *Information Sciences 212* (2012), 44–56.

[389] Peterson, W., and Brown, D. Cyclic codes for error detection. *Proceedings of the IRE 49*, 1 (1961), 228–235.

[390] Pielou, E. *An Introduction to Mathematical Ecology.* Wiley-Interscience, New York, 1969.

[391] Pielou, E. *Ecological Diversity.* Wiley, New York, 1975.

[392] Pitman, E. The estimation of the location and scale parameters of a continuous population of any given form. *Biometrika 30* (1939), 391–421.

[393] Prange, E. Cyclic error-correcting codes in two symbols. Tech. Rep. AFCRC-TN-57-103, Air Force Cambridge Research Center, Bedford, Mass., 1957.

[394] Puri, M. L., and Ralescu, D. A. Fuzzy random variables. *Journal of Mathematical Analysis and Applications 114*, 2 (1986), 409–422.

[395] Quesada, A. Monotonicity and the Hirsch index. *Journal of Informetrics 3*, 2 (2009), 158–160.

[396] Quesada, A. More axiomatics for the Hirsch index. *Scientometrics 82* (2010), 413–418.

[397] R Development Core Team. R*: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria, 2015. http://www.R-project.org.

[398] Rademaker, M., and De Baets, B. A threshold for majority in the context of aggregating partial order relations. In *Proc. 19th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'10)* (Barcelona, Spain, 2010), IEEE, pp. 1–4.

[399] Rademaker, M., and De Baets, B. Aggregation of monotone reciprocal relations with application to group decision making. *Fuzzy Sets and Systems 184*, 1 (2011), 29–51.

[400] Rademaker, M., and De Baets, B. A ranking procedure based on a natural monotonicity constraint. *Information Fusion 17*, 1 (2014), 74–82.

[401] Rajagopalan, S., and Schulman, L. J. Verification of identities. *SIAM Journal on Computing 29*, 4 (2000), 1155–1163.

[402] Reiser, R. H., Bedregal, B., and Baczyński, M. Aggregating fuzzy implications. *Information Sciences 253* (2013), 126–146.

[403] Rényi, A. On the dimension and entropy of probability distributions. *Acta Mathematica Hungarica 10*, 1–2 (1959), 193–215.

[404] Ristad, E. S., and Yianilos, P. N. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*, 5 (1998), 522–532.

[405] Rojas, K., Gómez, D., Rodríguez, J. T., and Montero, J. Some properties of consistency in the families of aggregation operators. *Advances in Intelligent and Soft Computing 107* (2012), 169–176.

[406] Ronkainen, T., Oja, H., and Orponen, P. Coputation of the multivariate Oja median. In *Proc. Intl. Conf. Robust Statistics* (2003), pp. 344–359.

[407] Rousseau, R. Woeginger's axiomatisation of the h-index and its relation to the g-index, the $h(2)$-index and the r^2-index. *Journal of Informetrics 2*, 4 (2008), 335–340.

[408] Rousseeuw, P. J., and Hubert, M. Regression depth. *Journal of the American Statistical Association 94*, 446 (1999), 388–402.

[409] Rousseeuw, P. J., and Ruts, I. Algorithm AS 307: Bivariate location depth. *Applied Statistics 45* (1996), 516–526.

[410] Rousseeuw, P. J., and Ruts, I. Constructing the bivariate Tukey median. *Statistica Sinica 8* (1998), 827–839.

[411] Rousseeuw, P. J., Ruts, I., and Tukey, J. W. The bagplot: A bivariate boxplot. *The American Statistician 53*, 4 (1999), 382–387.

[412] Rousseeuw, P. J., and Struyf, A. Computing location depth and regression depth in higher dimensions. *Statistics and Computing 8* (1998), 193–203.

[413] Rousseeuw, P. J., and Struyf, A. Computation of robust statistics: depth, median, and related measures. In *The Handbook of Discrete and Computational Geometry*, J. E. Goodman and J. O'Rourke, Eds. Chapman & Hall/CRC, Boca Raton, 2004, pp. 1279–1292.

[414] Rousseeuw, P. J., Van Aelst, S., and Hubert, M. Regression depth: Rejoinder. *Journal of the American Statistical Association 94*, 446 (1999), 419–433.

[415] Rousseuw, P. J., and Ruts, I. The depth function of a population

distribution. *Metrika 49* (1999), 213–244.

[416] Rubin, D., and Little, R. *Statistical Analysis with Missing Data*. John Wiley & Sons, 2002.

[417] Ruts, I., and Rousseeuw, P. J. Computing depth contours of bivariate point clouds. *Computational Statistics & Data Analysis 23* (1996), 153–168.

[418] Sanchez, D., and Trillas, E. Measures of fuzziness under different uses of fuzzy sets. In *Proc. IPMU 2012 (CCIS 298)* (2012), S. Greco et al., Eds., Springer-Verlag, pp. 25–43.

[419] Schönherr, S. *Quadratic Programming in Geometric Optimization: Theory, Implementation, and Applications*. PhD thesis, Swiss Federal Institute of Technology, Zurich, Switzerland, 2002.

[420] Schumaker, L. *Spline Functions: Basic Theory*. Cambridge University Press, 2007.

[421] Schweizer, B., and Sklar, A. *Probabilistic Metric Spaces*. Elsevier, Amsterdam, 1983.

[422] Shannon, C. A mathematical theory of communications. *Bell System Technical Journal 27*, 3 (1948), 379–423.

[423] Shao, J. *Mathematical Statistics*. Springer, New York, 2007.

[424] Shilkret, N. Maxitive measure and integration. *Indagationes Mathematicæ33* (1971), 109–116.

[425] Simovici, D., and Jaroszewicz, S. An axiomatization of partition entropy. *IEEE Transactions on Information Theory 48*, 7 (2002), 2138–2142.

[426] Sinova, B., Ángeles Gil, M., Colubi, A., and Van Aelst, S. The median of a random fuzzy number. The 1-norm distance approach. *Fuzzy Sets and Systems 200* (2012), 99–115.

[427] Sinova, B., Casals, M., Colubi, A., and Ángeles Gil, M. The median of a random interval. In *Combining Soft Computing and Statistical Methods in Data Analysis*, C. Borgelt et al., Eds. Springer, 2010, pp. 575–583.

[428] Sinova, B., Gonzales-Rodriguez, G., and Van Aelst, S. An alternative approach to the median of a random interval using an l_2 metric. In *Synergies of Soft Computing and Statistics for Intelligent Data Analysis*, R. Kruse et al., Eds. Springer, 2013, pp. 273–281.

[429] Sinova, B., Pérez-Fernández, S., and Montenegro, M. The Wabl/Ldev/Rdev median of a random fuzzy number and statistical properties. In *Strengthening Links between Data Analysis and Soft Computing*, P. Grzegorzewski et al., Eds. Springer, 2015, pp. 143–150.

[430] Sklar, A. Fonctions de répartition à n dimensions et leurs marges. *Publications de l'Institut de Statistique de L'Université de Paris 8* (1959).

[431] Small, C. G. Measures of centrality for multivariate and directional distributions. *Canadian Journal of Statistics 15*, 1 (1987), 31–39.

[432] Small, C. G. A survey of multidimensional medians. *International Statistical Review 58*, 3 (1990), 263–277.

[433] Somervuo, P. J. Online algorithm for the self-organizing map of symbol

strings. *Neural Networks 17* (2004), 1231–1239.
[434] Springer, M. D. *The Algebra of Random Variables*. John Wiley & Sons, New York, 1979.
[435] Stephens, M. EDF statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association 69* (1974), 730–737.
[436] Stigler, S. M. Linear functions of order statistics. *The Annals of Mathematical Statistics 40*, 3 (1969), 770–788.
[437] Sugeno, M. *Theory of fuzzy integrals and its applications*. PhD thesis, Tokyo Institute of Technology, 1974.
[438] Sylvester, J. J. A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics 1* (1857), 79.
[439] Szmidt, E., and Kacprzyk, J. Distances between intuitionistic fuzzy sets. *Fuzzy Sets and Systems 114*, 3 (2000), 505–518.
[440] Tai, K.-C. Tree-to-tree correction problem. *Journal of the ACM 26*, 3 (1979), 422–433.
[441] Tellier, L.-N. The Weber problem: Solution and interpretation. *Geographical Analysis 4*, 3 (1972), 215–233.
[442] The CGAL Project. *CGAL User and Reference Manual*, 4.6 ed. CGAL Editorial Board, 2015.
[443] Torra, V. On some relationships between hierarchies of quasi-arithmetic means and neural networks. *International Journal of Intelligent Systems 14* (1999), 1089–1098.
[444] Torra, V. Learning weights for the quasi-weighted means. *IEEE Transactions on Fuzzy Systems 10*, 5 (2002), 653–666.
[445] Torra, V. OWA operators in data modeling and reidentification. *IEEE Transactions on Fuzzy Systems 12*, 5 (2004), 652–660.
[446] Torra, V. Aggregation operators and models. *Fuzzy Sets and Systems 156* (2005), 407–410.
[447] Torra, V. Information fusion. Methods and aggregation operators. In *The Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Springer, 2010, pp. 999–1008.
[448] Torra, V., and Narukawa, Y. The interpretation of fuzzy integrals and their application to fuzzy systems. *International Journal of Approximate Reasoning 41* (2006), 43–58.
[449] Torra, V., and Narukawa, Y. *Modeling Decisions: Information Fusion and Aggregation Operators*. Springer-Verlag, 2007.
[450] Torra, V., and Narukawa, Y. The h-index and the number of citations: Two fuzzy integrals. *IEEE Transactions on Fuzzy Systems 16*, 3 (2008), 795–797.
[451] Tukey, J. W. Mathematics and the picturing of data. *Proc. Intl. Congress of Mathematicians* (1974), 523–531.
[452] Ukkonen, E. On approximate string matching. *Lecture Notes in Computer Science 158* (1983), 487–495.
[453] Ukkonen, E. Approximate string-matching with q-grams and maximal matches. *Theoretical Computer Science 92* (1992), 191–211.
[454] van der Loo, M. The stringdist package for approximate string match-

ing. *The* R *Journal 6*, 1 (2014), 111–122.

[455] van Kreveld, M., Mitchell, J. S., Rousseeuw, P., Sharir, M., Snoeyink, J., and Speckmann, B. Efficient algorithms for maximum regression depth. *Discrete and Computational Geometry 39*, 4 (2008), 656–677.

[456] Vardi, Y., and Zhang, C.-H. The multivariate l_1-median and associated data depth. *Proceedings of the National Academy of Sciences 97*, 4 (2000), 1423–1426.

[457] Vintsyuk, T. Speech discrimination by dynamic programming. *Cybernetics 4*, 1 (1968), 52–57.

[458] Wagner, R. A., and Fischer, M. J. The string-to-string correction problem. *Journal of the ACM 21*, 1 (1974), 168–173.

[459] Wallis, W., Shoubridge, P., Kraetz, M., and Ray, D. Graph distances using graph union. *Pattern Recognition Letters 22*, 6–7 (2001), 701–704.

[460] Waltman, L., and van Eck, N. J. The inconsistency of the h-index. *Journal of the American Society for Information Science and Technology 63*, 2 (2012), 406–415.

[461] Wandelt, S., et al. State-of-the-art in string similarity search and join. *SIGMOD Record 43*, 1 (2014), 64–76.

[462] Wang, X., and Kerre, E. E. Reasonable properties for the ordering of fuzzy quantities (I). *Fuzzy Sets and Systems 118*, 3 (2001), 375–385.

[463] Warshall, S. A theorem on Boolean matrices. *Journal of the ACM 9*, 1 (1962), 11–12.

[464] Weber, S. Measures of fuzzy sets and measures of fuzziness. *Fuzzy Sets and Systems 13* (1984), 247–271.

[465] Weiszfeld, E. Sur le point par lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematics Journal 43* (1937), 355–386.

[466] Welzl, E. Smallest enclosing disks (balls and ellipsoids). *Lecture Notes in Computer Science 555* (1991), 359–370.

[467] Widrow, B., and Winter, R. Neural nets for adaptive filtering and adaptive pattern recognition. *Computer 21* (1998), 25–39.

[468] Wilkin, T., and Beliakov, G. Weakly monotonic averaging functions. *International Journal of Intelligent Systems 30*, 2 (2015), 144–169.

[469] Wilkin, T., Beliakov, G., and Calvo, T. Weakly monotone averaging functions. *Communications in Computer and Information Science 444* (2014), 364–373.

[470] Wilkin, T. A. *Weakly monotonic averaging with application to image processing.* PhD thesis, Deakin University, 2014.

[471] Winkler, W. String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In *Proc. Section on Survey Research Methods, American Statistical Association* (1990), pp. 354–359.

[472] Winkler, W. E. Overview of record linkage and current research directions. Tech. Rep. 2006-2, U.S. Census Bureau, Washington, DC, 2006.

[473] Woeginger, G. J. An axiomatic analysis of Egghe's g-index. *Journal of Informetrics 2*, 4 (2008), 364–368.

[474] Woeginger, G. J. An axiomatic characterization of the Hirsch-index. *Mathematical Social Sciences 56*, 2 (2008), 224–232.

[475] Woeginger, G. J. A symmetry axiom for scientific impact indices. *Journal of Informetrics 2* (2008), 298–303.

[476] Wooley, J. C. Trends in computational biology: A summary based on a RECOMB plenary lecture. *Journal of Computational Biology 6* (1999), 459–474.

[477] Yager, R. R. Quasi-associative operations in the combination of evidence. *Kybernetes 16*, 1 (1987), 37–41.

[478] Yager, R. R. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man, and Cybernetics 18*, 1 (1988), 183–190.

[479] Yager, R. R. Connectives and quantifiers in fuzzy sets. *Fuzzy Sets and Systems 40* (1991), 39–75.

[480] Yager, R. R. Fusion of ordinal information using weighted median aggregation. *International Journal of Approximate Reasoning 18* (1998), 35–52.

[481] Yager, R. R., and Kacprzyk, J., Eds. *The ordered weighted averaging operators. Theory and applications*. Kluwer Academic Publishers, Norwell, 1997.

[482] Yager, R. R., Kacprzyk, J., and Beliakov, G., Eds. *Recent Developments in the Ordered Weighted Averaging Operators*. Springer, 2011.

[483] Yager, R. R., and Rybalov, A. Uninorm aggregation operators. *Fuzzy Sets and Systems 80* (1996), 111–120.

[484] Yager, R. R., and Rybalov, A. Understanding the median as a fusion operator. *International Journal of General Systems 26*, 3 (1997), 239–263.

[485] Yan, J. Enjoy the joy of copulas: With a package copula. *Journal of Statistical Software 21*, 4 (2007), 1–21.

[486] Yang, Q. The PAN-integral on the fuzzy measure space. *Fuzzy Mathematics 3* (1985), 107–114.

[487] Yianilos, P. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proc. ACM-SIAM Symp. Discrete Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1993, pp. 311–321.

[488] Zadeh, L. A. Fuzzy logic = computing with words. *IEEE Transactions on Fuzzy Systems 4*, 2 (1996), 103–111.

[489] Zadrożny, S., and Kacprzyk, J. Computing with words for text processing: An approach to the text categorization. *Information Sciences 176* (2006), 415–437.

[490] Zeng, W., and Li, H. Inclusion measures, similarity measures, and the fuzziness of fuzzy sets and their relations. *International Journal of Intelligent Systems 21* (2006), 639–653.

[491] Zhang, D. Triangular norms on partially ordered sets. *Fuzzy Sets and Systems 153* (2005), 195–209.

[492] Zhang, K., and Shasha, D. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*

18, 6 (1989), 1245–1262.

[493] Zuo, Y. Projection-based depth functions and associated medians. *The Annals of Statistics 31*, 5 (2003), 1460–1490.

[494] Zuo, Y., and Serfling, R. General notions of statistical depth function. *The Annals of Statistics 28*, 2 (2000), 461–482.

Index

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

PhD STUDIES

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

The Project is co-financed by the European Union from resources of the European Social Fund

ISBN 978-83-63159-20-7