JYVÄSKYLÄ STUDIES IN COMPUTING
63

Sami Äyrämö

Knowledge Mining Using Robust Clustering

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 63

Sami Äyrämö

Knowledge Mining Using Robust Clustering

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi Villa Ranan Blomstedtin salissa
marraskuun 17. päivänä 2006 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
In the Building Villa Rana, Blomstedt Hall, on November 17, 2006 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2006

Knowledge Mining
Using Robust Clustering

JYVÄSKYLÄ STUDIES IN COMPUTING 63

Sami Äyrämö

Knowledge Mining Using Robust Clustering

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2006

Editors
Tommi Kärkkäinen
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN:9513926559
ISBN 951-39-2655-9 (PDF)

ISBN 951-39-2621-4 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2006

ABSTRACT

Äyrämö, Sami
Knowledge Mining using Robust Clustering
Jyväskylä: University of Jyväskylä, 2006, 296 p.
(Jyväskylä Studies in Computing
ISSN 1456-5390; 63)
ISBN 951-39-2655-9
Finnish summary
Diss.

This work is devoted to the development of scalable and robust algorithms for data mining and knowledge discovery problems. The main interest lies in so-called prototype-based clustering methods that are implemented using iterative relocation algorithms. Different elements of prototype-based data clustering are discussed and basic algorithms are described. In order to support the usability of the new methods and algorithms, a modified knowledge mining process model is also proposed. The refined model is based on the well-known knowledge discovery process, but it emphasizes more domain analysis and "black box" nature of data mining. Significance and importance of knowledge mining are clarified by outlining the current body of the existing knowledge with real applications.

As the main outcome of this thesis, a highly automated robust clustering method is presented. The method consists of a number of separately developed and tested elements such as initialization, prototype estimation, and missing data strategy. Non-smooth nature of the robust statistics is rigorously considered from the point of view of non-smooth optimization. Numerical and statistical properties, such as robustness, scalability, computational and statistical efficiency, of the presented methods are tested and illustrated through a number of numerical experiments. The results are completed with some analytic results and illustrative real-world examples. Furthermore, in order to estimate the correct number of clusters, a new proposal of a cluster validity index is given.

Keywords: data mining, knowledge discovery, knowledge mining, data clustering, robust estimation, non-smooth optimization, visualization

Author

Sami Äyrämö
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Supervisors

Professor Tommi Kärkkäinen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

PhD Kirsi Majava
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers

Professor Vladimir Estivill-Castro
School of Computing and Information Technology
Griffith University
Australia

Professor Gurusamy Arumugam
Department of Computer Science
Madurai Kamaraj University
India

Professor Hideo Kawarada
Faculty of Distribution and Logistics Systems
Ryutsu Keizai University
Japan

Opponent

Academy Professor Heikki Mannila
Helsinki University of Technology and University of Helsinki
Finland

ACKNOWLEDGEMENTS

First of all, I want to express my sincere gratitude to Professor Tommi Kärkkäinen, who has acted as the principal supervisor for this thesis and introduced the interesting field of knowledge mining to me. His trust, support, scientific expertise and an immense reservoir of ideas made it possible for me to accomplish this work. Moreover, I have been delighted by his friendship that has supported me in the middle of all hard work and rush. I am also grateful to my other supervisor Dr. Kirsi Majava for her scientific support in all stages of this work. Besides her generosity and expertise in commenting, discussing, and advancing this work, Kirsi has also been a great friend to me. I also thank Dr. Erkki Heikkola for inspiring and encouraging me to start postgraduate studies.

I wish to thank Professor Vladimir Estivill-Castro, Professor Gurusamy Arumugam, and Professor Hideo Kawarada for reviewing the manuscript and providing me with insightful comments. I also want to thank Steve Legrand for checking the language of the manuscript.

A special thank goes also out to Professor Pekka Neittaanmäki, Director of Agora Center, for his encouragements to proceed and advance with this undertaking. With him I also thank all the staff at Agora Center, of whom I want to especially mention Päivi Fadjukoff, Päivi Parikka, Virva Nissinen, and Esa Kannisto.

For financial support I want to thank Agora Center and the Department of Mathematical Information Technology (both at the University of Jyväskylä), the Finnish Funding Agency for Technology and Innovation (TEKES), the Centre of Expertise Programme (OSKE), and the Ellen and Artturi Nyyssönen Foundation. I am also thankful to Jyväskylä Science Park, Metso Paper, UPM, TeliaSonera, and TietoEnator for their fruitful collaboration and funding.

During the ten years that I have studied and worked at the University of Jyväskylä, I have enjoyed the company of many great friends. My heartfelt thanks for the enjoyable moments and great spirit to all my great workmates at the Department of Mathematical Information Technology. Timo, Jonne, Sacha, Leena, Jaana, Turo, Paavo, Nieppa, and Olli, thank you for all the good time that you have shared with me during and outside working hours. Of my fellow students I want to especially thank Ossi, Tamu, Sampo, Mikko, and Pekka. Furthermore, I also wish to thank all of my friends and relatives, who are not directly related to this work, but have always been an important part of my life.

I think that parents can never hand the silver spoon to their children. Therefore, I see the true value of good parenting in supporting and guiding their children towards interesting but safe directions. I have been fortunate to grow up in such an extremely loving and supportive environment. Thus, the ones who definitely deserve my most heartfelt appreciation are my parents, Seija and Kari. Without their persistent and unselfish support, I would never have come this far in my life. I am also blessed to have a lovely sister, Terhi, who has always been very supportive of me and an important person in my life. I wish to thank also her sweet family for their support and kindness. The three great kids, Minea,

Lumi, and Jimi, deserve my special thanks for making my life so much richer and happier during the last three years.

Finally, I want to express my appreciation to my dearest lady Santtu. Her love, patience and understanding, especially during the final half a year of the writing process that often made me work round-the-clock, has been incredible and priceless to me. For this and for thousands of other reasons, I love You!

Jyväskylä
October, 2006
Sami Äyrämö

NOTATIONS AND ABBREVIATIONS

$\mathbb{R}^p$	p-dimensional Euclidean space
$\mathbb{N}$	set of natural numbers
ε, λ	scalars
i, j, k	positive integers $\{1, 2, 3, \ldots\}$
$\mathbf{u}, \mathbf{v}$	(column) vectors
$\mathbf{x}_i$	ith element in the set of vectors $\{\mathbf{x}_i\}$
$\{\mathbf{x}_i\}_{i=1}^n$	set of (column) vectors
$(\mathbf{v})_i$	ith component of a vector $\mathbf{v}$
$\mathbf{X}$	$n \times p$ data matrix
x_{ij} or $(\mathbf{X})_{ij}$	element of matrix $\mathbf{X}$ in row i of column j
NaN	not a number
$\mathbf{v}^T$	transpose of a vector $\mathbf{v}$
$\mathbf{\Sigma}$	the sample covariance matrix
$\mathbf{I}_p$	$p \times p$ identity matrix
$\mathbf{W}$	within group dispersion matrix
$\mathbf{B}$	between group dispersion matrix
$\mathbf{T}$	total dispersion matrix
$\mathrm{tr}(\mathbf{W})$	trace of matrix $\mathbf{W}$
$\det(\mathbf{W})$	determinant of matrix $\mathbf{W}$
$N_p(\boldsymbol{\mu}, \mathrm{I}_p)$	p-variate normal distribution with mean $\boldsymbol{\mu}$ and covariance I_p
$L_p(\mathbf{a}, \mathbf{b})$	p-variate Laplace distribution with center $\mathbf{a}$ and scatter $\mathbf{b}$
K	number of clusters
$\mathbf{m}_i$	$1 \times p$ vector of a cluster prototypes
$\{\mathbf{m}_i\}_{i=1}^K$	a set of cluster centers
$\mathbf{c}$	$\mathbb{N}^n$ vector of cluster assignments
n_k	number of points in cluster k
$\mathcal{C}_k$	k^{th} cluster
maxit	maximum number of iterations
t	iteration counter
$\mathcal{J}(\mathbf{x})$	objective function value at $\mathbf{x}$
$\nabla\mathcal{J}(\mathbf{x})$	gradient of function $\mathcal{J}$ at $\mathbf{x}$
$\partial\mathcal{J}(\mathbf{x})/\partial x_i$	partial derivative of function $\mathcal{J}$ with respect to x_i
$\mathcal{J}'(\mathbf{x}; \mathbf{d})$	directional derivative of function $\mathcal{J}$ at $\mathbf{x}$ in the direction $\mathbf{d}$
$\mathcal{J}^o(\mathbf{x}; \mathbf{d})$	generalized directional derivative of function $\mathcal{J}$ at $\mathbf{x}$ in the direction $\mathbf{d}$
$\partial\mathcal{J}(\mathbf{x})$	subdifferential of function $\mathcal{J}$ at $\mathbf{x}$
$\boldsymbol{\xi} \in \partial\mathcal{J}(\mathbf{x})$	subgradient of function $\mathcal{J}$ at $\mathbf{x}$
$C^m(\mathbb{R}^p)$	the space of functions $f : \mathbb{R}^p \to \mathbb{R}$ with continuous partial derivatives up to order m
$[a, b]$	closed interval from a to b
$]a, b[$	open interval from a to b
$\lvert\mathcal{I}\rvert$	number of elements in set $\mathcal{I}$

$\mathcal{O}$	time complexity or space requirement of algorithm
$\arg\min \mathcal{J}(\mathbf{x})$	point where function $\mathcal{J}$ attains its minimum value
$\mathcal{E}(\mathbf{X})$	expected (vector) value of $\mathbf{X}$
σ_j	variance of the j^{th} variable
$\sup M$	supremum of an ordered set M
$\inf M$	infimum of an ordered set M
$P(\mathbf{x})$	probability of $\mathbf{x}$
$\boldsymbol{\theta}$	statistical parameter
$\tilde{\boldsymbol{\theta}}$	estimate of a statistical parameter $\boldsymbol{\theta}$
$T(\mathbf{X})$	parameter estimator
$\boldsymbol{\mu}$	the sample mean
$\pi(n)$	permutation on $\{1, ..., n\}$
$diag(\lambda_1, \ldots, \lambda_p)$	$p \times p$ diagonal matrix
$B(\mathbf{x}, \delta)$	open ball with center $\mathbf{x} \in \mathbb{R}^p$ and radius $\delta > 0$, i.e. $\{\mathbf{y} \in \mathbb{R}^p \mid \|\mathbf{y} - \mathbf{x}\| < \delta\}$
$B_c(\mathbf{x}, \delta)$	closed ball with center $\mathbf{x} \in \mathbb{R}^p$ and radius $\delta > 0$, i.e. $\{\mathbf{y} \in \mathbb{R}^p \mid \|\mathbf{y} - \mathbf{x}\| \leq \delta\}$
DM	data mining
KD	knowledge discovery
KDD	knowledge discovery in databases
KM	knowledge mining
EDA	explorative data analysis
CDA	confirmatory data analysis
CG	conjugate gradient method
GS	golden section method
NM	Nelder-Mead method
SOR	successive overrelaxation
PCA	principal component analysis
MDS	multidimensional scaling
ICA	independent component analysis
RAM	random access memory
BIC	Bayesian information criterion
MDL	minimum Description Length
MCCV	Monte Carlo cross-validation
pdf	probability density function
ROI	return on investment
SOM	self-organizing map
MAD	median absolute deviation
CRM	customer relationship management
TOA	technology opportunities analysis

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

This thesis considers a new and interdisciplinary field of computer science and information technology called *data mining* (DM) and *knowledge discovery* (KD) [118, 115]. *Knowledge discovery in databases* (KDD) originated in the late 1980s from an observation that the growth rate of new knowledge lags behind the growth rate of data collection [316]. Since then the rate of data growth has accelerated further and, currently, large data sets are common in most (especially data-intensive) organizations. For example, business, industry, and government systems gather up data on an ongoing basis. Furthermore, many research organizations possess masses of data for scientific purposes. These data sets create a challenge for information system experts, statisticians, etc. By providing methods for transforming data into an understandable form and turning it into useful knowledge, DM is of great assistance to these experts from those numerous fields allowing them to efficiently utilize these large and heterogeneous real-world data sets.

Overall, DM comprises methods and techniques from various fields [170, 167, 96, 369]. The most important ones of these fields are illustrated in Figure 1. The focus of this thesis is to provide a thorough coverage of those fields of DM that are relevant for the actual topic. There is a particular emphasis on statistics, numerical optimization, and visualization. Elements of functional analysis are used for verification of new techniques. Sometimes, the terms DM and KDD are used interchangeably. More often DM, in which algorithms and methods are applied to data, is considered to be the core step of the KDD process. In this thesis, KDD means a process and DM its subprocess. The owner of the KDD process is a domain expert, whereas DM is controlled by a method specialist. This division and a slightly modified process-based view (knowledge discovery as the domain process) is presented in Chapter 2 (cf. Section 1.1 also)

Data clustering, which is the target DM method in this thesis, is a descriptive data mining technique that is used for partitioning a data set in an unsupervised manner [9, 174, 8, 204, 220, 203, 106, 400]. It is often considered as one of the core methods of the DM and KDD field. The basic idea of data clustering is very simple: to divide objects into groups so that the objects in the same group are more similar to each other than objects in the other groups.

The major aim of this work is to develop, verify, and validate a new clus-

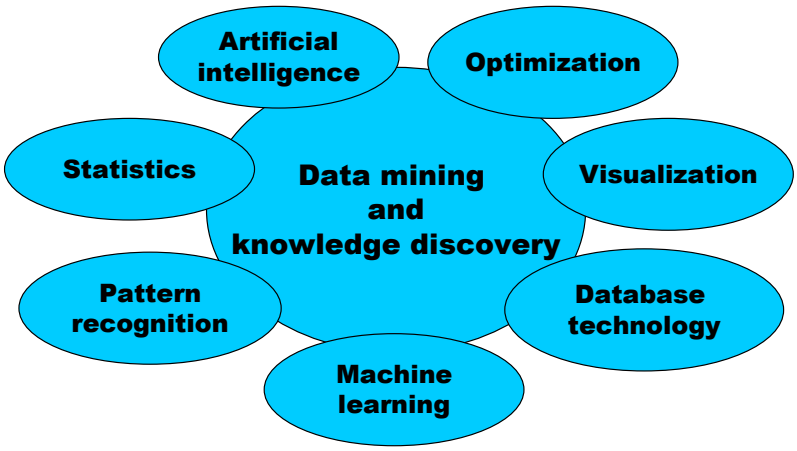


FIGURE 1 Origins of knowledge discovery and data mining.

tering method with several desirable properties. First, the method should be efficient and scalable even on large data sets. Secondly, it should be robust against erroneous and missing data values. Although many attempts to enhance the robustness of clustering algorithms have been proposed, not many of them are related to the field of DM. Overall, the algorithm should be highly automated so that the following operations and parameters would not involve the end users:

- Missing data handling
- Outlier pruning
- Initial parameter values
- Number of clusters

The existence of these types of clustering methods enables domain experts to mine knowledge with descriptive DM methods without thorough understanding of algorithmic and statistical details.

Generally, the contributions and lessons of this work involve the following interest groups:

- *Domain experts* should have basic understanding of the KM process and the division of activities between KDD and DM subprocesses. They should also know their domain and capabilities of DM tools, which, however, should be automatically applicable without, e.g., parameter tuning.
- *Statisticians* should know that robust M-estimates that result from non-smooth optimization problems can not be treated with classical C^1 calculus. Therefore, special analysis and numerical techniques are needed.
- *Optimization methodologists* should be aware of simple techniques that seem to be highly efficient for special non-smooth problems.

- *Information system specialists* should utilize the data that is collected and stored. Efficient retrieval is not enough.

- *DM methodologists* should remember the importance of rigorous analysis and testing of various methods. Moreover, with real data clusters can be retained better by using robust rather than classical projections. Special visualization techniques to support industrial process analysis can be developed.

1.1 Author's contributions

The outcome of this thesis is a result of an intensive collaboration between the author and the supervisors. Many of the initial ideas have been received from the supervisors, (especially the ideas of robust clustering using the SOR-based approach to the computation of the spatial median by Kärkkäinen, e.g., [213]). The ideas were further processed by the author, who has constantly exploited supervisors' in-depth expertise in the (sub-)fields of the topic, e.g., [214, 215]. Finally, through further analysis and method development work, that are supplemented with thorough computer experiments and literature studies, the author has advanced and extended the initial ideas to the fields of DM, KDD and KM by deriving, for example, new initialization and validation methods for cluster analysis. Overall, the author concludes this thesis with the following contributions:

1. Based on experiences from applied research projects, a new KM model for finding novel and useful knowledge from large data masses is proposed. The model consists of two merged parts that are based on the original KDD process and DM step. However, the new KM model presents the DM step as a subprocess of the KDD process. Although the main contributions of this thesis are concentrated on the DM parts of the KM model, the other steps are also considered, even if somewhat incidentally. Based on practical experiences and co-operation in applied industrial projects, the importance of domain analysis has been emphasized. Hence, an interesting idea of using the so-called genre method for a thorough domain analysis is introduced. This allows domain experts to better understand and be aware of the available information.

2. As a major contribution of this thesis, a new robust clustering method, that is built from thoroughly tested components, is developed. The development of these new methods encompasses the following contributions:

    - An extensive survey of elements of prototype-based data clustering is given.
    - The concept of robust clustering is described through the notions of non-smooth optimization.

- The general principle of prototype-based iterative relocation algorithms (e.g., K-means) is generalized for different norms, estimates, and missing data.
- Convergence of the general clustering algorithm with missing data treatment is shown.
- Using classical optimization methods and algorithms, performance of two types of robust clustering algorithms are tentatively examined and compared with the traditional K-means algorithm on erroneous and incomplete synthetic data sets.
- A new algorithm for computing a robust location estimate, the spatial median, on incomplete and noisy data sets is proposed.
- Following the ideas in [213], the problem of the spatial median is presented using the theories of non-smooth optimization.
- Based on ideas in [281], a simplified proof for the existence and uniqueness of the spatial median is given.
- A smoothed formulation for the problem of the spatial median with a missing data treatment is given.
- Convergence properties of the new algorithm for computing the spatial median are analyzed.
- Thorough numerical and statistical experiments for the spatial median estimator with the missing data treatment are presented.
- Based on the robust statistics and missing data treatment, an existing sub-sampling based cluster refinement algorithm is generalized for obtaining a consistent initialization method for the clustering problem.
- The proposed robust initialization and a "trimmed" variant are compared to several original and modified methods through extensive numerical experiments.
- Based on the so-called silhouette index, a new and robust validity index for the clustering problem is introduced.
- Based on robust scatter matrices and the missing data treatment, robust projection techniques are introduced and examined for visualizing high-dimensional data clusters in DM applications.
- Graphical techniques for visualizing the progress (fluctuations in process state) of an industrial process are proposed.
- Utility of the robust clustering algorithm with the robust validity indices, supplemented with robust projection and visualization techniques is demonstrated on real-world data sets.

1.2 Structure of the dissertation

The introduction is given in the first chapter. Chapter 2 discusses the DM and KDD process models and presents the basic idea and structure of the KM model. A genre-based strategy for the domain analysis step is suggested. Moreover, the utility and needs of KM are presented in an extensive literature survey on DM and KDD applications.

Chapter 3 provides a thorough discussion on data clustering. The main elements of the clustering algorithms are explained. For instance, the dissimilarity measures and missing data mechanisms are discussed and a number of known strategies for dealing with missing data are considered. Furthermore, a general iterative relocation clustering principle is introduced. Several existing variants of the well-known K-means clustering method are presented with algorithms.

In Chapter 4, the principles of non-smooth optimization and robust statistics are presented. A couple of basic optimization methods and the basic terminology of classical statistics are introduced. The requirements and measures of robust statistics are also presented with some illustrative examples. An in-depth discussion is given for a class of statistical location estimators known as M-estimators. Two robust and multivariate special cases of this class are the spatial median and the coordinate-wise median. The existing connection between M-estimation and non-smooth optimization is clarified by presenting non-smooth mathematical formulations and giving a strict analysis (incl. a simplified proof of existence and uniqueness of the solution in the problem of the spatial median) for the estimators from both statistical and computational point of views.

In Chapter 5, robust clustering methods, built on the aforementioned robust estimators, are tested on synthetic data sets. Prior assumption about the gains of robustness and utility of the chosen missing data strategy in the data clustering is assessed through numerical experiments on erroneous and incomplete data sets. A convergence analysis for the special cases of the general iterative relocation algorithm is given with a chosen missing data strategy.

Chapter 6 introduces a pair of new ”accelerated” methods for solving the non-smooth problem of the spatial median. Smooth formulations for the problem of the spatial median are given by taking into account the chance of missing data values. The computational efficiency and reliability of the new iterative methods are compared to several methods including, for example, classical gradient optimization methods and direct-search optimization methods. The theoretical values of efficiency and consistency of the new methods are considered from the statistical point of view. Comparisons are also made about how the chosen missing data strategy effects the statistical efficiency of the estimates.

In Chapter 7, new initialization methods with ”trimmed variants” for the problem of data clustering, that is generally known to be non-convex by nature (i.e., several locally optimal solutions exist), are introduced by exploiting the proposed robust estimators. A thorough examination against existing initialization principles is performed for the proposed methods.

In Chapter 8, the practical utility of the proposed tools is demonstrated on large real-world data sets. Before the examples, a general exploration through the cluster validity methods is given and on this basis, a new robust validity index for the problem of estimating the ”best” number of clusters, is introduced. By combining the proposed index to the advancements of the previous chapters, a new and highly automated robust clustering method is introduced. For comparison, the estimates for the number of clusters are computed by using the new robust indices and an old one. For interpretation of the results, robust scatter estimates are introduced and applied to the data projection. Based on the existing results, three techniques, one based on classical and two on robust statistics, are used for computation of the so-called principal directions of the data that are then used for visual interpretation of the results. The gains of the robust principal directions for clustering application on large data sets are then assessed.

In chapter 9, the results of the previous chapters are concluded and future needs and ideas are considered.

2 FROM KNOWLEDGE DISCOVERY AND DATA MINING TO KNOWLEDGE MINING

2.1 What is data mining and knowledge discovery?

> *"The workshop confirmed that knowledge discovery in databases is an idea whose time has come."*

This was stated during the first KDD workshop in 1989 [316], which served as a kind of kick-off for the data mining and knowledge discovery discipline. During the following years, the one-day workshop grew into a series of KDD conferences. The need for KD methods emerged from the development of digital data acquisition and storage systems during the last couple of decades. Capabilities in various organizations to utilize huge data storages, for example, in decision support or process control tasks, had lagged behind the growth rate of the data. As Bradley [42] states, the growth of knowledge in organizations has been outstripped by the growth of available data. On the other hand, Kohavi et al. [233] remind us that modern information technology provides good opportunities to build systems that take data mining and knowledge discovery issues into account in advance.

The term data mining was related to the knowledge discovery in the 1990s. It was related to the use and development of the algorithms in KDD problems. Many of the DM methods and techniques have already been in use in other disciplines. However, the fast development in the field of information technology has produced new requirements and the old methods have been developed further. Hence, the ideas behind DM and KD have been derived by researchers from several fields, such as statistics, pattern recognition, software engineering, machine learning, artificial intelligence, database technologies, and many others [316] (see Figure 1). Clearly, DM and KDD have an interdisciplinary background. Perhaps the most influential disciplines have been statistics and machine learning. The relation between statistics and DM is discussed, for example, in [354, 172, 171, 131]. Note that unlike statistics, DM is not based on data collection strategies. Thus it is sometimes referred to as *secondary data analysis* [170, 171]. Database technology

has also become an important element, because data must be stored and accessed effectively. The database issues are considered, e.g., in [127, 201, 69].

Another very important issue is visualization, and, in particular, provision of tools for visual interpretation of the obtained results. For example, scatter, trellis, and star plots can be used for visualization of multidimensional data [170]. Methods, such as *principal component analysis* (PCA) and *multidimensional scaling* (MDS), exist for dimension reduction tasks [170, 175]. Additionally, optimization and numerical mathematics skills are needed to develop fast and accurate methods for model fitting.

> *"Data mining is at best a vaguely defined field; its definition largely depends on the background and views of the definer."*

As the above definition, written by Friedman [131] in 1997, attests, DM has been regarded as an immature discipline. These definitions vary a lot depending on the background of the person expressing her/his views. This is not surprising when considering how short is the history of computer science and information technology and the fact that DM/KDD follows the progress of these fields. The relation of DM and KDD is discussed, e.g., in [117]. Margaret Dunham [96] separates DM and KDD by referring to the following short definitions:

Knowledge discovery in databases is the process of finding useful information and patterns in data.

Data mining is the use of algorithms to extract the information and patterns derived by the KDD process.

On the other hand, Hand et al. [170] define DM as: "

> *Data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner."*

Common definitions often consider DM as a step of the KDD process in which algorithms are applied, but as it will be later explained, DM can be also considered as a nested process for KDD. DM is associated with numerous different labels, such as data analysis, pattern analysis, knowledge extraction, information discovery, exploratory data analysis, database exploration, data pattern processing, information harvesting, siftware, data dredging, snooping, fishing, data archaeology, etc. [170, 167, 96, 318, 144, 60]. If one wants to quibble, "data mining" might be considered as an illogical label; for example, Han and Kamber [167] prefers to replace it by ''knowledge mining from data". They explain this through an analogy to gold mining. When gold miners are digging gold from sand and rocks, it is called gold mining. When a data analyst is mining knowledge from data, it would be logical to call it knowledge mining. However, since data mining is a well-established term, it will be used here, but later in the thesis,

a new aggregation of KD and DM will be proposed, and referred to as knowledge mining (KM)[1].

The amount of data is growing in two ways [116]. First, the number of collected objects n (or observations, records, etc.) is increasing. It is not unusual to have $n \gg 10^4$ objects in a data set. For example, an industry process can be controlled by measuring its state every 1/10 seconds over many years. Secondly, the number of attributes p is also growing as more and more sensors and measurement equipments are available for process control systems, or more and more high-dimensional special DM applications, for example, text and web mining, become available. As the capability to measure and record facts continuously grows in many fields, many old-fashioned data analysis techniques run into the so-called "*curse of dimensionality*" [170, pp. 193–196]. Traditional data analysis tools cannot cope with the existing large multidimensional data sets.

Many data stores contain thousands of attributes. For example, our research groups' experience in the process industry indicates that the number of simultaneous measurements recorded from the paper making process is in several hundreds. When such data is digitally stored over a couple of years, the database easily can contain tens of thousands of observations. Such masses of data, rather than assisting the domain experts or analysts in data exploitation, can overwhelm them and, as a consequence, the data stores become useless. Therefore, it is no longer enough to investigate data sets by looking at trends of individual measurements or observations. In real-life situations the information may be present in many forms, such as patterns, associations, changes, anomalies, or other significant structures. Furthermore, the data itself can exist in many formats and in different storage forms such as, text, image or sound files, databases, data warehouses, and many others (cf. the categories of communication forms (CCF) in [378]). Alongside with the increasing computing power, efficient algorithms are needed for accessing and making sense of the most substantial information lying in the data masses.

As most of the data analysis methods have originally been designed only for ideal cases with a feasible amount of error-free data, a number of fundamental research topics, such as efficiency of the algorithms, significance of the domain knowledge, handling of uncertainty, etc., have been addressed since KDD-89 workshop. During the last two decades, numerous promising methods for analysis of large data sets have been developed and integrated by DM/KKD researchers and engineers. The aforementioned problems, along with future issues, such as handling of complex data, relevance of visualization and perceptual presentation, have remained in the main focus of knowledge discovery field until today.

DM puts a colorful set of data analysis and modelling techniques under one umbrella. In addition to the curse of dimensionality and other problems that have emerged due to large data sets, other classical problems, such as the "*bias-variance dilemma*", are still looking for solutions [170, p.223]. For example, in the

[1] The term knowledge mining is also used by Professor Ari Visa, see the site of Knowledge Mining course at http://www.cs.tut.fi/ avisa/8004202.

case of data clustering, this means that too large number of clusters does not abstract the data enough and the solution remains difficult to interpret, whereas too small number of clusters may lead to a biased solution with a significant loss of information. In regression and classification problems, one has to avoid *overlearning*, i.e., that a classifier learns an individual data set too precisely. Such a classifier does not generalize outside the learning data set. Before one can choose the most appropriate methods, the type of information needs to be analyzed and understood especially from the application domain perspective.

Prior to deeper and more detailed treatments of the topic, definitions for data, information and knowledge are given. According to Stenmark [359] and Tuomi [377], the following definitions serve as guidelines for this thesis:

- Data consists of not yet interpreted symbols, such as simple facts, measurements, or observations.
- Information consists of structured data with meaning.
- Knowledge emerges from information after interpretation and association with the context.

2.1.1 Data mining tasks

Hand et al. [170] define the following DM tasks:

Exploratory Data Analysis (a.k.a. EDA) means explorative analysis of a data set where the goal is to visually observe interesting and unexpected structures. The visual exploration is realized by using graphic representation techniques, such as histograms, pie charts, scatter plots etc. High dimensional data sets, that is $p > 3$, are difficult to visualize and dimension reduction techniques, such as PCA and MDS can be used to transform the data into a low-dimensional space (e.g., [170]).

Descriptive modelling is used to describe a high-dimensional data set in a refined way without strong assumptions about underlying classes and structures. This is also called *unsupervised classification*. Cluster analysis, segmentation, density estimation, and dependency modelling techniques are applied for this purpose. Descriptive cluster models are the major issue of this thesis.

Predictive modelling consists of classification and regression. These are also referred to as *supervised classification*. The aim is to predict a value of a particular variable from the known values of other variables. In classification, the predicted variable is categorical (e.g., the diagnosis of a disease), whereas in regression the variable is quantitative (e.g., the price of stocks). Building of predictive models relies on prior knowledge about classes and structure in data. For instance, a neural network classifier is built by using a learning data set where each data point has a known class

label [181]. By applying a learning algorithm to the network, a deterministic mapping from data points to the classes is obtained. The best network is the one which is precise enough for the learning data but also generalizes to other data sets. Depending on the application, the prior knowledge about classes may be collected from domain experts or by investigating the data with explorative and descriptive techniques. Predictive DM techniques include, among others, neural networks, nearest-neighbor techniques, decision trees, and Bayesian techniques. The methods for predictive data mining mostly originate from the fields of machine learning and statistics (see, e.g., [94]).

Discovery of patterns and rules aims at finding interesting and uncovered relationships from large data sets. The information is presented by association rules and sequential patterns. Market basket analysis is a traditional example of the applications in this category. Another good example is the analysis of telecommunication networks' alarms [229]. The seminal association rule algorithm was proposed by Agrawal et al. [4] in 1993. Since then many faster variants have been introduced, e.g., [5].

Retrieval by content aims at finding patterns of interest from large data sets. This is utilized especially for documents or images from large sets. For example, a user may have a set of keywords, an image, a piece of music, or just a description of an image or song, and she/he wants to find a set of documents that matches best with the interesting patterns. WWW search engines are retrieval-by-content tools, of which Google (www.google.com) is an excellent example.

When considering this classification of DM tasks from a statistical perspective, it can be challenged. Mainly it is a question about the role of explorative data analysis (EDA) with respect to DM as a whole. Among the statistics community there exist two types of data analysis: explorative and confirmatory data analysis, e.g., [188]. The principles and procedures of confirmatory data analysis (CDA) are considered as great intellectual products of the last century [374]. CDA produces summary statistics, assesses significance and precision, and tests hypotheses on data that is collected under strict and specific circumstances. Exact confirmation and reproducibility of the results are also important issues. This is accomplished by exact formulation of the hypothesis, strict experimental design, data collection, and analysis. Finally, one confirms or discards the hypothesis based on the outcome of the fixed assumptions and chosen methods.

While the exact confirmation of hypothesis is the basic principle of most scientific research, EDA has also a significant role, for example, in supporting hypothesis making. EDA is flexible, simple to apply, and a quick way to increase one's understanding about data. This is actually needed also for the steps of the CDA [375]. This means that the whole DM clearly resembles EDA. In DM one is not confirming or discarding hypotheses. Because a knowledge discoverer is searching for new and unexpected facts (such as groupings, dependencies, correlations etc.) from data, she/he is actually doing exploratory data analysis, at

least, if considering from the point of view of the aforementioned traditional definitions. Therefore, DM as a whole can be seen as a kind of explorative data analysis approach, which makes the name of the first category somewhat questionable. Hence, perhaps a better label for the first task is ”visualization based data exploration”.[2]

2.1.2 Components of data mining algorithms

A data mining algorithm is an instantiation of a particular method that actually performs some chosen operations on the data. Therefore, data mining algorithms are often presented as a composition of separate components that must be adapted for different tasks. Fayyad et al. [111] propose a three component model, whereas Hand et al. [170] suggest a four component framework with a data management strategy.

In the following, the main components of data mining algorithms according to [170, 111] are described.

A model or pattern structure determines an underlying structure or a functional form for the data. A model realizes a meaningful function (e.g., classification or clustering) through a representational form (e.g., linear function of multiple variables or Gaussian probability density function) [111].

A score function (a.k.a. cost function, loss function, criterion function, or goodness-of-fitness function) determines how the attained model fits the data set, and presents the error between the model and real-world data. The error (that is the value of the score function) is minimized by using optimization and iterative search methods. A score function in a regression problem might be, for example, the least-squares fit between the obtained function and data or, in a classification problem, the misclassification rate.

Optimization and search methods are used to optimize the score functions for finding the best-fitted models and pattern structures. The search methods are divided into two categories: parameter search for a given model and model search from a model space. Parameter search problems are usually formulated as optimization problems, such as the aforementioned minimization of the least squares error of a given regression model. The pattern/model search problems are accomplished, for example, by using heuristic search techniques (e.g., many clustering algorithms [106]). There are many challenges related to the model search and parameter optimization. The variance-bias dilemma comes up in this part of KDD. It means that models and methods should be designed so that too precise matching between a model and a particular data set is avoided. Otherwise the obtained models become overfitted, which prevents them to be generalized to other

[2] This discussion on the role of EDA in the fields of DM and KDD was initiated by professor Vladimir Estivill-Castro during the evaluation of this thesis.

cases. On the other hand, the methods should solve the problems accurately in a reasonable time, which, especially on large DM data sets, brings out the issue of computational complexity. Therefore, the development of efficient and accurate solvers for the model and parameter optimization is a challenging task, and is investigated mainly by the scientists in the field of optimization and numerical mathematics. Optimization methods that are useful in many DM problems are introduced, e.g., in [28, 311, 267, 280].

Data management strategy considers the efficient handling of the data during the pattern/model search/optimization step. For example, the development of the most classical statistical and traditional machine learning algorithms have relied on the assumption that all data can be stored in and accessed from the fast random-access memory (RAM). Data mining is, however, intended for large data sets that exceed the size of RAM. Therefore, the DM algorithms and data storage strategies must be designed to be scalable to large data sets, which means that careful consideration of data management strategy is needed.

This thesis concentrates mainly on the first three components by developing new formulations and algorithms for location estimation and clustering problems. However, real-world experience has shown that the data management issue can not be totally avoided, because of the huge size of the current data collections.

A large number of articles and books have been published on the methods and principles of DM. Perhaps the most extensive discussions from the methodological point of view are given in [170, 167, 96]. Data Mining and Knowledge Discovery journal[3] presents the latest developments. Furthermore, the techniques and methods employed by DM community can be found in books from many other fields, for example, in books about statistics, artificial intelligence, machine learning, pattern recognition, and database technology.

2.2 Knowledge discovery process

Although DM and KDD may often be considered as a set of computational and statistical methods for solving knowledge discovery problems, primarily they are iterative and interactive processes involving numerous steps from domain analysis to interpretation and utilization of the results. While DM is managed by computational experts, KDD is managed by a domain specialist. KDD starts from masses of data and proceeds, by using intelligent computational and statistical methods, to the refined presentation that clarifies the interesting information in exploitable and interpretable form using reduced amounts of numbers, graphs, rules, etc. Fayyad et al. [117] define the goals of the KDD process as a verification of a user's hypothesis, autonomous discovery of new patterns, prediction

[3] Data Mining and Knowledge Discovery is published by Springer Netherlands (http://www.springerlink.com).

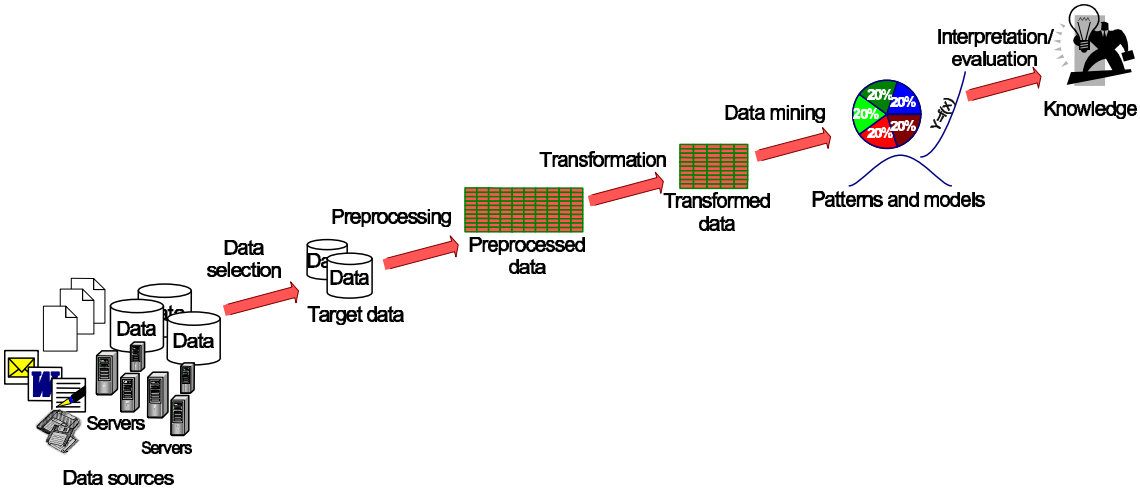


FIGURE 2 A typical KDD process model (modified from [115]).

of future behavior of some entities, and/or description of interesting patterns. By interpreting the refined and compressed information, a domain specialist can turn the discovered information into human knowledge and obtain the intended goals. A widely used and accepted definition for the knowledge discovery reads as

> *"The nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data"* [111].

The term *pattern* is an expression that describes a subset of data or a model suitable for the subset of the data [111]. For example, Andrassyova et al. [10] and Vehviläinen [385] summarize and compare different definitions and process models that are applied in KDD. Although the definitions somewhat differ from each other (see, e.g., [40, 350, 167, 111, 73]), in the main points they typically follow the definition that is given in many articles by Fayyad et al. (e.g., [111, 115, 116, 113, 117]). In the following, short descriptions for the main steps of the KDD process are given (cf. Figure 2):

Step 1. Data selection Interesting data from heterogeneous data sources (databases, files, etc.) are selected. The modified view in Figure 2 emphasizes the current situation, where the data sources exist in multiple heterogeneous formats. Therefore, the first step of the process is presented by a figure that fits the present state of information sources.

Step 2. Data preprocessing Erroneous and missing data values are handled, for instance, by imputation and outlier detection methods. The intention of this thesis is to minimize the effort needed in this step. This is realized by applying strategies that are tolerant against erroneous data values and, hence, allow the utilization of all available data.

Step 3. Data transformation The data is transformed into a suitable form by finding the most significant features and variables. Data reduction and projection techniques (e.g., PCA) are utilized.

Step 4. Data mining This is the core step of the KDD process. At this point, the algorithms are selected. Interesting models and patterns are extracted from the transformed data. Which methods are chosen depend on the application (cf. Section 2.1.1).

Step 5. Interpretation and evaluation The refined information is presented in understandable and useful ways to the user. Visualization and knowledge representation techniques are exploited. These are often exploratory DM techniques (cf. Section 2.1.1). This activity is performed in co-operation with domain specialist. Returning back to previous steps is possible.

These steps provide a useful outline for the overall KDD process. Fayyad et al. [111] give also an augmented version for the process model in which, for example, the analysis of the application domain is included.

2.3 Knowledge mining: an integrated process model

The industry-related applied research indicates that before the most valuable information and knowledge can be discovered, a lot of attention has to be paid to the domain analysis. In large business organizations, huge amounts of ”unknown” or ”unregistered” communication and information exist that should be discovered and even digitized first or, otherwise, the reasonable goal setting and knowledge utilization may become inaccurate and inefficient. Hence, an increased awareness about information and communication residing in target environments (e.g., business, industry, or government) can be useful while one is defining the goal for the KDD process. This awareness is obtained by ”mining” the organization.

Many KDD process models focus on data that is digitally stored and, therefore, immediately available, but this is not necessarily the common situation anymore, because a lot of information exist in various formats. Without a thorough domain analysis, a part of this information may remain out of reach. In order to boost the utility of the KDD process, a so-called *genre-based domain analysis* [223, 379, 320] can be made part of domain analysis prior to any data processing activities. This was successfully utilized in applied projects related to paper industry [211, 210, 212].

2.3.1 Genre-based domain analysis

As the organizations produce, use, and manage large amounts of data in multiple formats, such as databases, text and image files, photos, letters, A4 paper documents, faxes, audio records, etc., it may become laborious to recognize the most

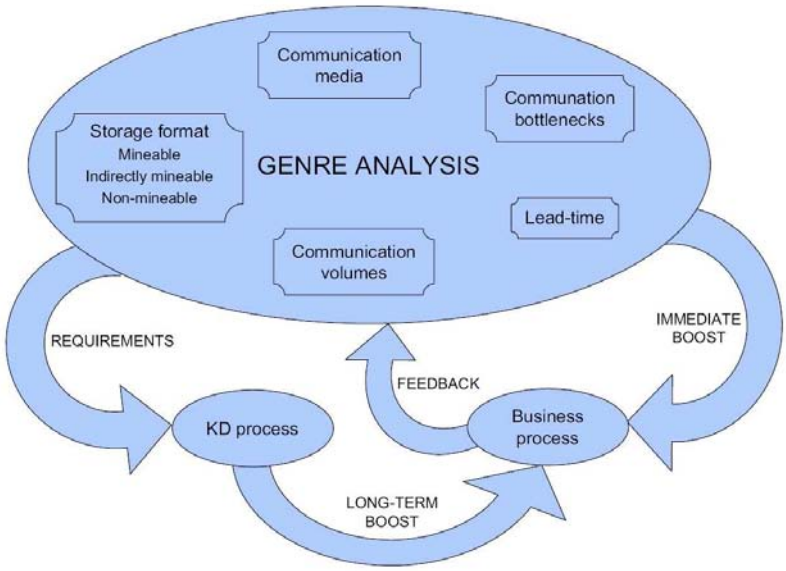


FIGURE 3 The domain analysis model.

important sources of information, or even be aware of their current owners. Some of the data formats, such as human data (i.e., tacit knowledge), letters, paper documents, or audio/videotapes, can not be directly utilized by the KDD process. Although such data often contain useful and valuable knowledge, it needs to be transformed into a utilizable form.

The concept of genre provides means for thorough domain analysis through examining information communicated as part of business processes, capturing all information flows [356] including verbal communication, data in information systems, and paper as well as electronic documents. In fact, genres can be thought of as prototypical models [368] for communication representing a typified piece of information responding to a recurrent communicative situation, and apart from individual's private motives [405, 404, 411].

The fundamental idea of using genres in domain analysis is the assumption that information that is communicated and used within business processes can be regarded as important, or even critical, information to the success of business operations and activities. Further, genre analysis provides knowledge over the most essential informational entities that are related to business processes. In this way, these entities can be taken into account in the KDD process in order to provide rational results. Thus, the goal of using the genre-based method [379] in this context is to map different data sources in the organization to each other and find out the related metadata, such as data amounts, formats, and users. Thereby, it supports the domain analysts in discovering and understanding the most important data and information sources in the organization. In this way, the existing sources are understood better and new valuable knowledge needs can be recognized. Furthermore, the type of data and information sources are understood and the most interesting non-digital information can possibly be digitized prior to the actual KDD activities.

Tyrväinen et al. [378] present a taxonomy for categories of communication forms that is used within the genre-based method. Based on this taxonomy, three kinds of data and information from the DM point of view can be defined:

Mineable data Digital data, such as bitmaps, XML-documents, Word documents, Excel sheets, e-mails, etc. These are relatively easily processed by computer systems.

Indirectly mineable data This resides, for example, in audio/VHS tapes or ordinary letters. Although not immediately ready for processing by computer systems, this type of data can still be digitized without content changes.

Non-mineable information Information that is delivered in rumors, discussions, negotiations, etc. In other words, it is ”human data” that is not mediated and nearly impossible to convert into a mineable data. Digitalization of this kind of information requires significant changes into the existing practices, information management systems, and workflows across the organization.

The first two classes can be considered as data types or storing formats, but the last one should be regarded as information rather than data. It is also very difficult to convert into data. Knowledge is an immaterial human property, which is based on the received information. This information need to be extracted from its owner and materialized in order to utilize it digitally. This is naturally an intractable problem and will not be considered further in this thesis. Besides the enhanced KDD activities, another advantage that is achieved by using the genre-based approach in business organizations, is the immediate feedback about the state and needs of the organization that the thorough domain analysis also provides. It produces valuable information from general process factors, such as the throughput-time of (sub)processes, managed information volumes per person, and the amounts of communication (see Figure 3).

2.3.2 Integration of DM and KDD processes

A modified KDD process including DM as its subprocess is presented in Figure 4. It follows the similar principles as the popular model proposed by Fayyad et al. [115] (see Figure 2), which is further specified in verbal descriptions in [115, pp.10–11]. Iterations back to the previous process steps are possible. However, because almost everything in today’s world is measured by *return on investment* (ROI), the original KDD model is extended with more domain specific steps and details. Our model starts from analysis of the present situation and ends up with intelligent utilization of data stores that is needed to make profit.

First of all, the knowledge discovery is divided into two nested processes: KD and DM. KD corresponds to the original KDD process and DM is an extension of the DM step embedded into KDD. As the process is extended to consider all kinds of mineable data and information sources, ’D’ for ”databases” is dropped from the KDD abbreviation. This aggregate will from now on be referred to as KM. By separating more clearly knowledge discovery from data mining, domain

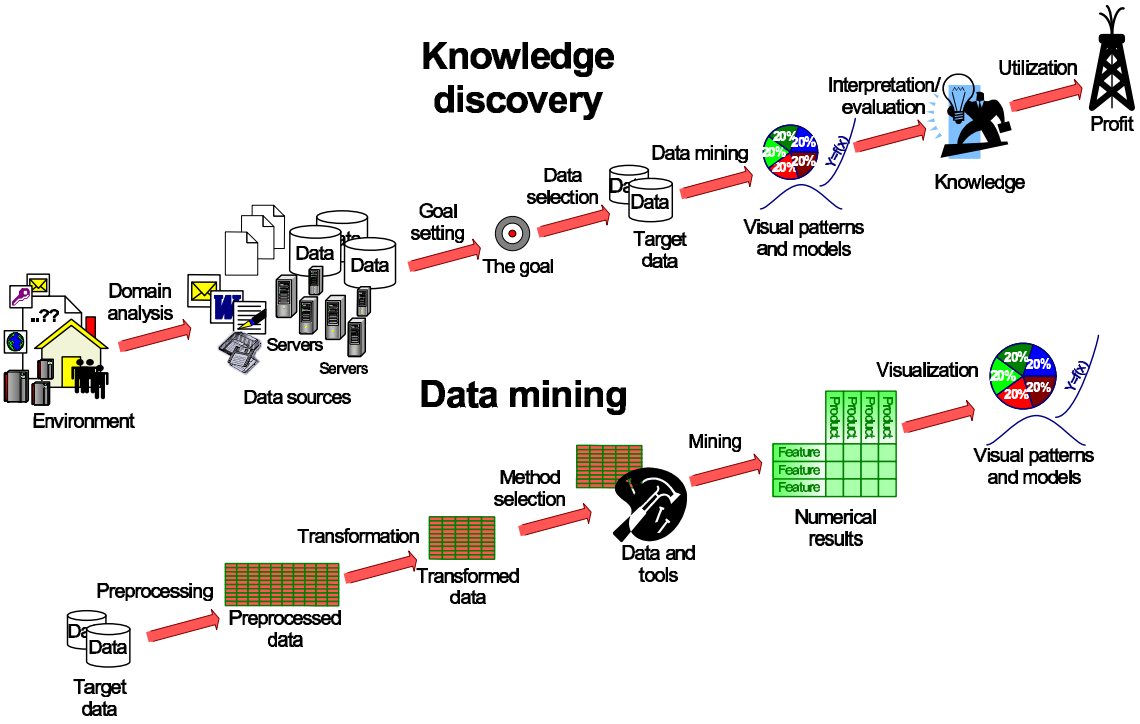


FIGURE 4 Knowledge mining: division of KDD into the KD and DM processes.

specialists are able to independently attend all steps of the knowledge discovery. Hence, DM is redefined as a sub-process that is hidden from the domain experts. It is desired to be highly automated, so that the domain experts can concentrate totally on the domain level problems. The original KDD process model in Figure 2 includes a number of complex steps, which means that for a successful pass through the process, one needs skills and knowledge from the organization issues to the computational methods applied, for example, in the feature selection or outlier detection. Therefore, it also presumes close co-operation between the domain expert and the method specialists. This is not necessarily an efficient approach in future as it necessitates the availability of DM support each time KD and DM operations are needed. Therefore, the data preprocessing and transformation steps should be included in the DM process. They should not concern the domain specialists.

The pictures representing the initial situation of the KD process in Figure 4 illustrate a typical start for a knowledge mining task. The figure emphasizes the fact that KM is always related to a target environment where it is to be realized. Some people know the environment better than the others: such information exist in the form of tacit knowledge and experience that accumulates in the course of time and is not easily transferred among people. The domain (background) knowledge is defined as *"the information not explicitly present in the data"* [316]. According to the workshop report [316], domain knowledge should be mainly used for reducing the search space, while it also should be utilized carefully, because it may prohibitively influence the discovery of unexpected models and patterns. In the KM approach, the main purpose of domain knowledge is to assist in the

search of the most useful data and information sources, and not only to explain the expected patterns or models in data. In the late eighties, the facilities for the analysis of the information sources or for collecting and digitizing data were meager than those available at present. Hence, the increased domain knowledge needs not be considered as a restrictive factor for any subsequent step of the KD or DM process. It rather assists in the adjustment of the goal and increases the capacity for the result utilization.

KD steps

The steps of the KD process are described next. Figure 4 shows the steps taken from the starting point to the profit stage. The arrows represent the steps and the symbols illustrate the outcomes. As the process is iterative, returning from any step to some of the previous steps is possible (cf. Section 2.2).

Step 1. Domain analysis A general domain analysis, as described in Section 2.3.1, is performed. One increases significantly his/her understanding about the target environment. It is important to be aware of the owners and the origin of the interesting data, whether all that data is in a digital format, about the needs to digitize data, of the availability of new and interesting information, and so on. On the other hand, one should evaluate the available resources required by the whole process. All these details effect the subsequent issues, such as data selection, goal setting, time constraints, etc.

Step 2. Goal setting As a result of the domain analysis, one is aware of the available data. On the other hand, there is awareness about when and by whom the data will be processed. For example, one may come up with situations where interesting data exist but where the only database expert happens to be unavailable. Hence, based on the outcome of the domain analysis one defines a reasonable goal for KM (cf. [385]).

Step 3. Data selection At this step one has gained technical readiness to make the data selection. She/he is aware of available resources and the reasonable goal. Based on knowledge about the useful information and communication sources, the most important data sets are selected. If the interesting data is not in directly mineable form, one should consider possible digitalization. This depends, for example, on human resources, facilities, and time constraints.

Step 4. Data mining In this step one performs data mining. The input to the data mining is the selected data and the output is information presented by understandable visual models and patterns. As the technical details of this step are hidden, computational or statistical skills by the domain specialist are not required. On the other hand, this excludes neither a human-driven nor automatic computer-driven approach to the data mining process. Hence, data mining can be performed by using a commercial DM software that

serves all DM activities or, it can be accomplished in a step-by-step fashion with a DM specialist.

Step 5. Interpretation/evaluation Interpretation and evaluation of the information presented by visual patterns and models should lead to increased knowledge by means of the predetermined goal. Depending on the domain, the knowledge means things like increased awareness of customer behavior, reasons for fluctuations in an industry process, relations and dependencies between medical treatments and health, etc. One should also integrate the obtained information to the background knowledge.

Step 6. Utilization "You'll get nothing by doing nothing...". In order to profit from the discovered knowledge, it has to be utilized. The value of the obtained knowledge does not show up until this real-life utilization is accomplished. Depending on the goal, it may mean an increased number of customers and sales for a company, survivals from a refractory disease, pre-empted terrorist attacks or prevented white-collar crimes and frauds. Moreover, this could also mean a decision to utilize KM more consistently for increasing business intelligence.

DM steps

Compared to the above KD steps, DM requires much deeper understanding on technical and computational issues (see, for example, algorithmic details in [185, 270, 44]). While the overall KD process is guided by a domain specialist, the DM step remains mainly hidden. End users are usually more keen to solve domain level problems than to struggle with computational or statistical details underlying the DM models and patterns (e.g., choice of metaparameters). For example, the mathematical and statistical elements used in DM tools are often so involved that a specialist is required to deal with them. Therefore, if many algorithmic decisions are required by the end user, usability of the whole KM may suffer significantly. Due to the aforementioned points, it is essential to automate the DM process as fully as possible. This means that after the goal setting and data selection, the more technical DM sub-process is performed as a black box system from the end users point of view.

The DM process consists of five steps, which merge together the technically complex steps of the original KDD process (cf. Figures 2 and 4). The process starts with masses of target data and finishes with the useful and understandable results. These steps are described next.

Step 1. Preprocessing In this step, the target data is preprocessed. According to a chosen strategy, missing values are replaced and erroneous values corrected or removed. If these activities are put in practice as part of the following mining algorithms, this step may not need any actions.

Step 2. Data transformation This step may involve scaling, feature selection, and dimension reduction operations.

Step 3. Method selection In this step, suitable methods for the task are selected (cf. Section 2.1.1). This depends on the data and the goal of the whole KM task.

Step 4. Mining This is the core of the KM. The chosen methods are applied to the data. The output can consist of clusters, association rules, class labels, estimates for parametric models, etc.

Step 5. Visualization The numerical results are converted into a useful and understandable form. Visualization techniques are used to illustrate the obtained structures and models.

As a result of this process, the data is refined and transformed so that domain experts can easily interpret and use it. Preprocessing and transformation, which both are kinds of preprocessing steps in the DM process, can overlap each other. Because the major goal of this thesis is to develop a clustering algorithm that minimizes the number of input parameters of the DM process, the above process framework is a focal point from the point of view of this thesis on the whole. It is desirable that a KD process framework based on this development, including the black-box approach to the DM process and automated reliable methods, should become more and more common in future, in order to allow growth in the utility, amount of users, and adoption of KD and DM tools.

2.4 Emerging areas: Text mining and Web mining

Special types of DM tasks, *text mining* and *Web mining*, can nowadays be considered such fundamental applications of DM and KM that they are worth of somewhat more detailed treatment. These specific sub-fields of KM offer many useful tools and techniques for companies dealing with e-commerce and customer relationship management (CRM) issues [39]. The special challenges have arisen due to the exponential growth rate of Internet currently producing enormous amounts of structured, semi-structured (e.g., html documents), and unstructured data. The unstructured data can be, e.g., informal text, image, audio and video files [317]. Consequently, the traditional query and data analysis tools have become inadequate. The unstructured data formats are troublesome for DM methods and tools. Hence, general-purpose markup languages, such as XML[4], are used to describe and store data in a more computer-friendly structured or semi-structured formats [73].

2.4.1 Text mining

Text mining focuses on information that is hidden in large collections of various text documents, for example, e-mails or web-pages [32, 133]. Real-world applica-

[4] In 9th of April 2006, First International Workshop on Knowledge Discovery from XML Documents (KDXD 2006) was held in Singapore.

tions consist of organizations' large document databases, such as help-desk data sets or generally useful spam-filtering systems. By text mining large document databases companies may obtain beneficial information about future technology opportunities, for example, to improve business strategy planning.

Mining from DM and KDD document collections

An interesting application from the point of view of this thesis is science and technology document database mining. DM and KDD, in particular, are excellent objects for text mining, because the field is progressing and, consequently, the number of publications is growing, at an unprecedented rate. This can be observed, for example, by taking a glance over the bibliography of this thesis. Due to interestingness of the used target data and the obtained results, a study published in 1999 by Zhu et al. [423] is briefly reviewed next. They propose a kind of text mining process that is called TOA (technology opportunities analysis) and apply it to mining scientific and technical document database INSPEC. Their main focus is on the documents from the field of data mining and knowledge discovery itself. The results provide valuable information about seminal contributions in the field, interaction patterns across separate fields and institutions, emphases and active players in the field, emerging research areas, etc. More generally, the output of the TOA process includes basic profiling of the R&D activity in the target area, mapping of technical topic interrelationships, and composite "science or technology indicators". The results provide very good examples about the possibilities and utility of text mining.

At first, they investigated how the terms DM and KDD occur in the articles. The results show that the institutions significantly contributing to KDD are almost the same as those contributing to DM. This may not surprise the DM community, but the authors note that, for example, in the fields of natural language processing and computational linguistics, the similar matching of contributors across the fields is not found even though these are conceptually very close fields. Hence, this is a sample of information that DM and KDD produce. The high frequency of co-occurrences of DM and KDD terms led to the consolidation of abstract of target documents that contained one or both of the terms. The results also show that multiple authorship is a common practice in DM/KDD (1142 authors for 694 articles). 27% of the contributors were from companies (the most prolific of them was IBM). Affiliations among the most prolific authors seem to follow the same division, since 22% of them are from companies and 74% from academic institutions. Hence, the case study suggests that the KDD/DM research is predominantly academic, but also supported by a significant industrial involvement. Based on the relatively larger number of conference papers against the journal papers and the large size of the research teams, the authors conclude that KDD/DM is a 'hot' research area with a lot of experimental and interdisciplinary application-oriented group work, where the results are disseminated quickly. Term clustering shows that 'deductive databases' and 'distributed databases' were the two most central terms of the clusters at that time. The au-

thors also found that 'World Wide Web' topic has the greatest average growth in the number of documents. 'Data warehouse' was the most industry-led domain, whereas 'rough sets', 'genetic algorithms', and 'pattern classification' were the most academic-driven. The slowest growth rates in the number of documents were found for 'pattern classification', 'genetic algorithms' and 'query processing'. The authors also note that the topics engaging industry researchers grew fastest. In addition to the two most industry-driven technical topics of 'data warehousing' and 'business process databases', also 'very large databases' and 'association rules' showed strong industrial participation, whereas the remaining technical topics were mainly academic. Hence, the development of basic approaches and techniques was predominantly addressed in the academic world in 1999.

2.4.2 Web mining

Alongside with text mining, web mining is another special application in the fields of DM and KD with a number of recent efforts [105, 235, 107]. Cooley et al. [77] define it simply as "*the discovery and analysis of useful information from the World Wide Web*." Web mining can be roughly divided into three categories [235, 107]: *Web content mining*, *Web structure mining*, and *Web usage mining*. Han et al. [166] give even a finer treatment for the Web mining tasks, but that will not be discussed here.

The web offers currently the most challenging data source for DM and KDD. Apart from that the content of the available on-line Web data is more complex than the content of traditional text document collections, its content is also much more dynamic and, worst of all, only a small fraction of the Web's pages contain truly relevant or useful information. This makes their use and finding the relevant information quite problematic for the Web users community, which consists of extremely broad spectrum of people with different backgrounds, interests, and purposes. Moreover, the Web's dynamic nature makes it highly unpredictable. Web sites' access patterns may change dramatically due to significant events, such as the WTC terror attacks on September 11, 2001 [166]. Leaving aside the random surfing guided by linkage pointers from one page to another, the current basic techniques for accessing Web information are mainly *keyword-based search* and *topic-directory browsing*. These types of facilities are provided, for example, by some well-known search engines such as Google and Yahoo.

E-commerce is a modern topic within business and an extremely interesting application for web mining techniques. Ansari et al. [13] state that "*E-commerce is the killer-domain for data mining*". As DM is inherently capable of dealing with a lot of electronic and mineable data, it provides good facilities for collecting and handling customer information from the Web [355]. Kohavi et al. [233] consider e-commerce as a promising field from the viewpoint of Web mining, since the data collection systems there are often new and do not carry much load from the past and ancient legacy systems, where the needs of data mining were not taken into account. So, Web mining offers good facilities for e-commerce traders to extract

useful information for essential purposes, such as market strategy planning, from the huge amounts of transaction and server data.

Web content mining

Web content mining concentrates on raw information available in the Web [107]. The Web contains several types of data, such as textual, image, audio, video, metadata, and hyperlinks. Mining of the aforementioned heterogenous data types is also referred to as multimedia mining (e.g., [167]), which is actually considered as an instance of Web mining in [235]. Mining of unstructured text data is better known as text mining (see Section 2.4.1), but Kosala et al. [235] see it as an instance of Web mining as well. The semi-structured data available on the Web can be refined with a higher level organization that enables the use of data mining techniques, such as intelligent search agents, database techniques or query systems, so that the users can more easily find relevant and interesting information from the Web.

Web structure mining

The Web consists, not only of huge numbers of pages, but also of links between pages that contain enormous amounts of information about the authority of pages etc. [62]. The analysis of these dependencies and connections is referred to as Web structure mining, and may provide a lot of valuable information, for example, about the authority of a Web site. This helps the search engines to direct users to the most useful pages. Content-based search strategies are too vulnerable for all kinds of misinterpretations.

Google A search engine, that is worth of few words in this thesis, is Google[5]. It is highly popular and a successful example of the *retrieval-by-content* type of data mining tools that also exploit linkage information. Google was developed by Sergey Brin and Larry Page [52] in the 1990s and, currently, 150 million searches are performed on Google daily[6]. Unlike traditional document search engines, finding of relevant pages by Google is not only based on counting the number of keywords occurrences. In order to suggest the most relevant and authoritative pages to users, Google employs an intelligent technology called PageRank, which evaluates more thoroughly the importance and authority of pages (see, e.g. [53, 310]). The relevance of page content is evaluated by Hypertext-Matching Analysis and Google analyzes the full content of a web page, including font size, location of words, and content of the neighboring web pages.

Web usage mining

Web usage mining [358] focuses on browsing and access patterns (e.g., clickstreams) generated by Web users. Such information about page access frequen-

[5] http://www.google.com/

[6] http://www.google.com/ads/pharma.html, accessed 17th of May, 2006.

cies or common traversal paths through a Web site is useful and valuable for e-commerce business, CRM issues, or Web-service providers. In order to provide users with better services, Web pages and links should be updated according to users' needs and to reflect the current trends. Information about Web usage is often presented in the form of association rules, sequential patterns, or cluster structures.

In addition to the usual challenges such as robustness, reliability and scalability of the DM methods, the Web brings further requirements. For example, in the context of clustering Web visitors that is actually closely related application to the topic of this thesis, Estivill-Castro and Yang [103] mention the problems of finding the natural measure of similarity for Web-visitors, scalability with respect to both the size of data and computation of the similarity measure, and quality of results in the presence of noise and outliers. For instance, when applying classical prototype-based clustering algorithms such as K-means for grouping users' navigation paths, the usual estimates for the cluster representatives will not necessarily provide useful information. This occurs because the navigation paths are discrete structures and the sample mean as a prototypical vector of values for a cluster may not have the values of a representative item of the cluster. In order to avoid this problem a medoid-based robust clustering algorithm, in which each cluster representative is restricted to be a member of the target data set, is developed for Web usage mining [103].

Using the developed methods, many Web-sites are nowadays customized, and some of them are even continuously adapted, according to the users' interests. This special area of Web usage mining is called *Web-personalization*. The aim is to produce personalized portals that dynamically serve customized contents for the users [286]. Vast amounts of information about the behavior of Web users is continuously stored and available in weblogs [107]. The problem is that usually only the visited URLs are stored. As such, URLs are quite a poor source of information, because they do not provide information about actual content of the pages. Consequently, weblogs are currently enriched also with content-based information [107]. At present, the amount of information stored into web-logs is huge (see examples in [306]). Because the amount of this information also grows continuously, the efficiency of the existing data mining algorithms lags behind the volumes of real world data. Therefore, more efficient web mining innovations also emerge. For example, in order to avoid dealing with the whole data each time that DM is required, stream-based data mining agents [306] store and exploit previous summary information. Intelligent agents provide a flexible framework for distributed Web mining, such as retrieval, filtering, and categorizing of web documents [346]. SUGGEST 2.0 [349] is an example of a recommendation system. It exploits Web usage mining techniques to dynamically generate suggestions about pages that have not yet been visited but might be of interest to a user. Lawrence et al. [246] have investigated a possibility to integrate DM tools with a recommender and remote shopping system. The basic idea of the system is a customer using Personal Digital Assistant (PDA) to compose and transmit the shopping list to a store. The system also targets personalized product rec-

ommendations back to the customer. A lot of DM techniques were applied, such as association mining to discover relationships between the product classes and clustering methods to group customers with similar shopping histories.

Generally speaking, Web mining is not anymore an added-value tool for e-business companies, but rather a new-found requisite in their efforts to attract and serve customers from among the enormous masses of users that are currently reachable via Internet. Business developers have become aware that due to the mediating nature of e-business, the customers are just as easily lost as reached. Kohavi et al. [232] remind us: *"leaving the store is only one click away..."* that well illustrates the ease by which Web-shopping can be cancelled. Internet provides no practical opportunities for face-to-face human interaction or sales-talks by a clever sales-man in a shopping situation. These have been the main tricks to make a customer to change her or his mind in a traditional shopping situation. Electronic stores are open round the clock and all the customers, within the technical limits of course, are able to arrive and do the shopping simultaneously. On a top seller Web-marketing site there can be an enormous number of customers visiting at the same time. After figuring out all these risks, problems, challenges and, especially, after paying attention to the fact that the Web contains an enormous amount of information about customer profiles, e-business companies have started to utilize Web mining more and more. In this way, they have become more intelligent in selecting the best pop-up advertisement, e-commerce offer, or the best-price buying suggestion for a particular customer [317, 193]. Hence, the knowledge about customers' needs and desires is one of the most significant competitive factors for any company on the e-commerce field. Consequently, as the e-business companies continuously gather and produce more and more data and, thereby, interesting problems for data miners, DM provides, therefore, a great advantage for the business party, such as amazon.com and eBay among many others [231, 233].

As an extremely promising subarea of DM and KDD fields, the coalition of e-business and data mining has received a lot of attention in scientific journals. Detailed descriptions about the techniques and results can be found, e.g., from a survey by Pierrakos et al. [319] and articles by Nasraoui et al. [296, 295, 294]. A special issue devoted to Web mining is provided by Data Mining and Knowledge Discovery journal (Volume 6, Number 1, January 2002). More about e-commerce applications can be found, for instance, in articles published in a special issue of Data Mining and Knowledge Discovery (Volume 5, Numbers 1-2, January 2001) journal.

2.5 Some application areas

In this section, DM/KDD applications and environments are surveyed more thoroughly. The examples have been collected from a diverse set of articles and books. For instance, a large number of clustering applications are introduced by Everitt

et al. [106]. The data sets used contain dolphin whistles, populations based on different factors such as economy, geography or climate, mammal's milk, and so on. These all can be analyzed with DM methods. Detailed analyzes of application-specific issues will not be considered here. Instead, the intention is to map and summarize the wide utility of DM. The set of applications overviewed is not expected to be complete, but, at least, can provide an idea about the vast potential of DM for an interested reader. As an alternative for this chapter, a quick browsing through the Internet would reveal the same thing.

Scientific applications Huge amounts of data in many forms, such as images, time series, and sequence data, are collected by scientific applications [110]. DM can assist scientists to discover and understand more deeper facts and meanings from these data sets. For example, some well-known knowledge discovery authors, among them Fayyad et al. [110] and Han et al. [165], introduce a number of emerging scientific domains and applications for DM.

Rocke et al. [334] propose an efficient method for star/galaxy classification from sky survey data by using sub-sampling and mixture likelihood clustering methods. Using various DM methods, such as PCA, K-means clustering, and *self-organizing maps* (SOM), Kitamoto [228] introduces a typhoon clustering application for a huge archive of 34,000 satellite images. Fayyad et al. [110, 114] present five DM case studies on scientific applications. SKICAT is an astronomical sky survey application [112], and was developed to identify classes of the sky objects appearing in photographic images by using decision-tree algorithms and to generate a catalog of them for astronomers. JARtool [57] was developed to help planetary geologists to recognize volcanoes on the surface of the planet Venus from huge amounts of high-resolution image data that was collected by a synthetic aperture radar (SAR) from the Magellan spacecraft for more than five years. The image database is huge, since it consists of more than 30,000 SAR images. A case study on mining astronomical time series is introduced by Ng et al. [301].

Fayyad et al. [110] refer to several gene-finding programs and methods that are applied to biosequence databases. Ressom et al. [330] propose an adaptive double self-organizing map (ADSOM) method for data clustering and visualization in gene expression applications. Biological data sets are inherent objects for DM systems because nature is full of different categories and hierarchies. In collaboration with several others Ka Yee Yeung [407, 409, 408, 406, 252] has written several articles about bioinformatics. The articles introduce several applications of clustering techniques, such as Bayesian mixtures, in gene expression. SUBDUE, introduced by Cook et al. [76], is a system for discovering interesting substructures from structural data, for example, protein and DNA databases.

CONQUEST is a parallel computer system for atmospheric scientists for making queries about extra-tropical cyclones and distinctive blocking features in the atmosphere [304]. Han et al. [165], for their part, outline, with discussions, several emerging scientific domains, such as telecommunication, climate and ecosystems, for data mining. All these scientific domains produce masses of

high-dimensional data in many forms. Han et al. [169] have also developed a spatial data mining prototype system called Geominer for geospatial data mining. Spatial data consists of objects with annotations about physical location information [96]. Quakefinder is a DM system for automatic detection and measurement of tectonic activity in the Earth's crust from satellite images [362].

Business and industry Economy and business domains belong to the most promising areas for DM methods (e.g., [14, 234]). Today's business managers are dealing with huge volumes of data that can be processed and refined by DM applications. Basic applications on the business field are, e.g., market basket analysis, risk management, and customer segmentation [350]. In order to maintain and even raise the competitiveness on today's market fields, companies apply DM in sales, marketing, supply chain optimization, and fraud detection [234]. For example, companies need to sense and forecast changes in the market trends. They also have to direct advertising campaigns to the right group of customers (e.g., [227]). Such customers can be found by customer segment analysis, in which customers are partitioned into homogenous groups according to their purchasing behavior or other demographic features [350, 170]. For example, Dolnicar et al. [88] have applied several clustering algorithms to travel markets segmentation. In the experiment on market segmentation problems by Hruschka et al. [189], neural networks outperformed the K-means algorithm.

Another interesting business application for DM is stock market analysis. The stock markets produce immense amounts of time-series data that can be segmented, clustered, classified, etc. in order to predict future behavior (see, e.g., [221]). Hence, one may search similarly behaving stocks and try to predict their future changes. For example, Gavrilov et al. [139] have studied the methods for finding groups of similarly varying stocks.

Modern telecommunication networks process vast amounts of transactions on a daily basis [343]. This data hides many useful patterns and regularities that can be discovered by DM methods. In cooperation with four Finnish telecommunication companies, Mannila et al. have developed methods to enhance the use of network alarm data [229, 178]. They applied association rule algorithms for mining frequent episodes from sequential data [271]. The increased knowledge was used for filtering redundant alarms, locating faults, and predicting severe faults. In [229], the same algorithms are also generalized into other environments, such as document collections and student enrollment data. A data mining software TASA (Telecommunication Alarm Sequence Analyzer) was implemented in order to realize the practical use of the methods [230]. Another example of using intelligent methods within the telecommunication industry is given by Weiss et al. [394]. Vehviläinen [385] introduces three DM methods (rough sets, CART classification trees, and SOM) as tools for managing quality of service in digital telecommunication networks. C4.5 decision tree classifier for a telecommunication network diagnosis application is presented by Danyluk and Provost [84]. Data warehousing and sequence mining from telecommunication network data in fault forecasting tasks and improvements in network reliability are introduced

in [343].

Aviation safety is an important issue, which is considered by Nazeri et al. [299]. They investigated the possibilities to assist safety officers in their analysis work and, as a result, developed a new DM tool called 'Aviation Safety Data Mining Workbench'. Aircraft service issues are also treated with DM methods. Létourneau et al. [248] developed an approach to predict aircraft component replacement using on-board sensor data and several DM techniques.

Corney [81] applies clustering methods to intelligent food design. The research objective was to find best method to help the food designers discover homogeneous groups of consumers, and, thereafter, target them with appropriate products. The work is an interesting example from the intersection of three research domains: product design, food and drink, and intelligent data analysis. As a result, a combination of a clustering and outlier detection algorithm, which aims at producing outlier-free cluster models, is introduced.

Spam Spam, that is unsolicited e-mails, have become a significant problem for e-mail users. Hence, spam filtering can be seen as a type of text mining task. Without spam filters, users would have to use an unacceptable amount of time for finding relevant messages amongst the spam. Further, spam e-mails misuse a lot of network resources and, perhaps, even prevent relevant messages from being received in time. An e-mail message contains numerical, categorical, and unstructured information. The numerical information may include the message length and the number of recipients. The categorical information can consist of the domain of the sender and the type of the attached files. Unstructured information is found on the subject and content fields. As the spam messages are in some sense similar to each other, the spam detection problem can be approached from the clustering perspective. Since the content and topic of e-mails usually change over time, online clustering can be used to classify the received messages into right classes. Manco et al. [268] apply text preprocessing and data clustering to the spam problem. By integrating an agglomerative hierarchical clustering algorithm with the k-means clustering, they obtained good results with respect to a user-defined "optimal" partitions. As a future task, they suggest extending the e-mail clustering to attachment file contents.

Fraud detection and risk prediction While organizations are storing increasing amounts of (possibly sensitive) data, the detection of frauds, intrusions, or other ominous and abnormal behavior becomes even more difficult. At the same time, it becomes more essential to maintain the viability of, for instance, payment systems. Therefore, data mining techniques are applied more and more, for example, to credit card fraud detection problems [63]. The compliance and integrity of the U.S. government crop insurance program was improved by using log-linear analysis to mine anomalous behavior that may indicate possible collusion between farmers, policy sellers, and indemnity claim adjusters [257]. Fawcett et al. [109] apply rule learning to uncover fraudulent behavior among cellular phone users. DM techniques, such as association rules and segmentation, have been ap-

plied by Viveros et al. [391], to fraud detection in health insurance information systems. In the Australian government's health insurance scheme, *Medicare*, DM was applied to discover valuable features from transaction data [180], and the obtained knowledge was used for predicting compliance in pathology laboratories. Lee et al. [247] outline DM-based research on real-time intrusion detection systems. Utility of data clustering in intrusion detection problems is considered by Portnoy et al. [322], who present a case study about intrusion detection, by single linkage clustering, in a military networks environment. DM is also used to predict the risk of an applicant in property and casualty insurance business: for example, the K-means clustering algorithm is applied to predict risk and claim frequency levels for automobile drivers [153].

Miscellaneous applications The integration of business and sport has brought along more money and performance enhancing technology into sport. Be it motor sports, endurance sports, team sports, or nearly any of the today's sports, a lot of digital data is collected. In endurance sports, huge number of measurements, such as heart rate, lactic acid level, or oxygen uptake, is collected continuously. According to the Web site of the McLaren F1 team[7], during the 2004 season they collected 40 gigabytes of race data and 75 gigabytes of test-drive data. In many team sports especially, immense amounts of game statistics (shots, misses, rebounds, passes, etc.) are collected. For example, the NHL ice-hockey league[8] is known for its numerous game statistics that have been collected over years. Hence, team sports such as basketball, ice hockey, and soccer, together provide yet another interesting field of research for DM applications.

Advanced Scout is an example of the sport DM software [36]. It has been utilized by a number of coaches in the professional NBA basketball league[9]. The software assists the coaches to discover interesting patterns from game data, which, in turn, helps them to plan game strategies for the coming matches. In order to remove errors from the data, the software performs a rule-based preprocessing step first. The data can also be optionally enriched with extra input data by domain experts. After the cleaning step, the data is transformed and reformatted. In the transformation process, the discrete events are grouped into possessions. Data may be further enriched with the roles of the players. This is done by using the inference rules and an additional data entry. During the DM step, the team coach initiates queries and the software detects interesting relationships within, for example, shooting performance or team possessions. The DM step algorithm is known as Attribute Focusing (AF). The discovered knowledge is presented both in text and graphic forms.

Digitalization of music has brought along many challenges, such as Web marketing and plagiarism detection, for data mining. Finding the most representative parts of a song helps one to compare different songs to each other. Mörchen et al. [287] introduce a multivariate time-series method for finding typical parts

7 http://www.mclaren.co.uk/

8 http://www.nhl.com

9 http://www.nba.com/

from songs. A compression-based method for hierarchical clustering of music is presented by Cilibrasi et al. [72]. The method is based on compression of strings that represent the music pieces. Similarity in music pieces is discovered using ordinary compression techniques. Music analysis methods enable more effective navigation in music databases and, for example, can serve as a recommender system for e-customers. The same approaches can be applied in other specific areas too, for example, in detecting plagiarism in student's computer programming assignments.

3 INTRODUCTION TO PROTOTYPE-BASED CLUSTERING METHODS

Data clustering, by definition, is an exploratory and descriptive data analysis technique, which has gained a lot of attention in statistics, data mining, pattern recognition, etc. It is used for unsupervised investigation of multivariate data sets with possibly different data types. These data sets also differ from each other in the number of objects and dimensions. Undoubtedly, data clustering belongs to the core methods of data mining, in which one focuses on large data sets with unknown underlying structure. In particular, since the beginning of the KDD era in 1989 great efforts to develop scalable methods for clustering large data sets have been made and, as a result of these efforts, several algorithms currently exist, e.g., CLARA (1990) [220], CLARANS (1994) [302], DBSCAN (1996) [98], BIRCH (1997) [419], STING (1997) [392], DBCLASD (1998) [401], IncrementalDBSCAN (1998) [97], GDBSCAN (1998) [342], K-modes and K-prototypes algorithms (1998) [192], PDBSCAN (1999) [402], CHAMELEON (1999) [217], CACTUS (1999) [137], ROCK [151] (2000), scalable EM-algorithm (2000) [41], CURE (2001) [152], and CLOPE (2002) [403]. This chapter is intended as an introduction into the most important issues of data clustering. Furthermore, the principles of clustering algorithms based on iterative location strategy are explained.

3.1 What is cluster analysis?

Cluster analysis is an important element of exploratory data analysis. It is typically directed to study the internal structure of a complex data set, which can not be described solely through the classical second order statistics (the sample mean and covariance). It is also called unsupervised classification, because class labels (a.k.a. response variables) are initially unavailable for the data. MacQueen [265] stated in his seminal paper that instead of being merely a computational process to produce a unique and definitive grouping for a given data set, data clustering is an aid to improve qualitative and quantitative understanding of large multivariate data sets. Later, due to its unsupervised, descriptive, and summarizing

nature, data clustering has also become a core method in data mining and knowledge discovery. Especially during the last decade, the increasing number of large multidimensional data collections have stepped up the development of new clustering algorithms [167, 170, 369].

Generally speaking, classification of different things is a natural process for human beings. There exist numerous natural examples about different classifications for living things in the world. For example, various animal and plant species are the results of unsupervised categorization processes made by humans (more precisely, domain experts), who have divided objects into separate classes by using their observable characteristics [135]. There were no labels for the species before someone generated them. A child classifies things in an unsupervised manner as well. By observing similarities and dissimilarities between different objects, a child groups those objects into the same or different group.

At the time before the computers came available, clustering problems were solved manually. Although it is easy to visually perceive groups from a two- or three-dimensional data set, such "human clustering" is not a very consistent procedure, since different individuals see things in different ways. The measure of similarity, or the level and direction one is looking at the data, is not consistent between different individuals. By direction is meant the set of features (or combinations of features) that one exploits in classification. For example, people can be classified into a number of groups according to their economical status or their annual alcohol consumption, etc. These groupings will not necessarily capture the same individuals [106]. The way the user looks at the data set depends, for example, on her/his background (position, education, profession, culture, etc.). It is clear that these things vary greatly among different individuals [204].

Numerous definitions for cluster analysis have been proposed in the literature. The definitions differ slightly from each other in their emphasis on the different aspects of the methodology. In one of the earliest books on data clustering, Anderberg [9] defines cluster analysis as a task aiming to "*find "natural groups" from a data set, when little or nothing is known about the category structure*". Bailey [16], who surveys the methodology from the sociological perspective, states that "*cluster analysis seeks to divide a set of objects into a small number of relatively homogeneous groups on the basis of their similarity over $\mathcal{N}$ variables*". $\mathcal{N}$ is the total number of variables in this case. Moreover, Bailey notes that "*conversely variables can be grouped according to their similarity across all objects*". Hence, the interest of cluster analysis may be in either grouping of objects or variables, or both (see also [106, p.154-155]). On the other hand, it is not rare to reduce the number of variables before the actual object grouping, because the data can be easily compressed by substituting the correlating variables with one summarizing and representative variable. From the statistical pattern recognition perspective, Jain et al. [203] define cluster analysis as "*the organization of collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity*". Hastie et al. [175] define the goal of cluster analysis from their statistical perspective as a task "*to partition the observations into groups ("clusters") such that the pairwise dissimilarities between those assigned to the same cluster tend to*

be smaller than those in different clusters". Tan et al. [369] state from a data mining point of view that "*cluster analysis divides data into groups that are meaningful, useful, or both.*". By meaningful they refer to clusters that capture the natural structure of a data set, whereas the useful clusters serve only as an initial setting for some other method, such as PCA or regression methods. For these methods, it may be useful to summarize the data sets beforehand.

The first definition emphasizes the unknown structure of a target data set, which is one of the key assumptions in cluster analysis. This is the main difference between clustering (*unsupervised classification*) and classification (*supervised classification*). In classification the category structure is known a priori, whereas cluster analysis focuses on data sets, where the class labels are unknown. Jain et al. [205] suggest that the class labels and all other information about data sources influence the result interpretation but not the cluster formation process. On the other hand, domain understanding is often of use during the configuration of initial parameters or correct number of clusters.

The second and third definitions stress the multi-dimensionality of data objects (observations, records, etc.). This is an important notion, since grouping of objects that possess more than three features can not attained by a normal human being without automated methods. Naturally, most of the aforementioned definitions address the notion of similarity. Similarity is one of the key issues of cluster analysis, which means that one of the most influential elements of cluster analysis is the choice of an appropriate similarity measure. The similarity measure selection is a data-dependent problem. Anderberg [9] does not use term "similarity", but instead he talks about the degree of "natural association" among objects. Based on the aforementioned definitions and notions, the cluster analysis is summarized as "*analysis of the unknown structure of a multidimensional data set by determining a (small) number of meaningful groups of objects or variables according to a chosen (dis)similarity measure*". In this definition, the term meaningful is understood identically with Tan et al. [369].

Even though visual perception of data clusters works up to three dimensions, in spaces consisting of more than three dimensions it becomes a complex approach. Therefore, computers are indispensable for multidimensional cluster analysis tasks. We know that a human is inconsistent as a classifier, but also different algorithms produce different groupings even for the same data set. Hence, there exists no universally best clustering algorithm [8, 205]. On this basis, Jain et al. [205] advise one to try several clustering algorithms when trying to obtain the best possible understanding about data sets. Based on the authors' experience and theoretical considerations, Kaufman et al. [220] propose six clustering algorithms (*PAM, CLARA, FANNY, AGNES, DIANA*, and *MONA*) that they believe to cover a major part of the applications. PAM is a partitioning-based *K-medoid* method that divides the data into a given number of disjoint clusters. CLARA, which also partitions a data set with respect to medoid points, scales better than PAM to large data sets, since the computational cost is reduced by sub-sampling the data set. FANNY is a fuzzy clustering method, which gives a degree of memberships to the clusters for all objects. AGNES, an agglomerative hierarchical

clustering method produces a tree-like cluster hierarchy using successive fusions of clusters. The method provides a solution for different numbers of clusters. DIANA is also a hierarchical method, but it proceeds in an inverse order with respect to AGNES. At the beginning, DIANA puts all objects into one cluster and continues by splitting each cluster up to two smaller ones at each step. MONA is also a divisive algorithm, but the separation of objects into groups is carried out by using a single variable. The set of methods, which was just presented, should give a reasonably overall view to the internal structure of any data set. As mentioned earlier, the result interpretation step is a human process, in which one may utilize different visualization techniques (e.g., PCA and MDS [170]). After the interpretation, prior domain knowledge and any other problem related information are integrated in the clusters.

The development of clustering methods is very interdisciplinary. Contributions have been made, for example, by psychologists [283], biologists [373, 206], statisticians [125], social scientists [16], and engineers [204]. Naturally, various names for cluster analysis have emerged, e.g., numerical taxonomy, automatic classification, botryology, and typological analysis [220, p.3]. Also unsupervised classification [369, 203], data segmentation [175], and data partition [331] are used as synonyms for data clustering. Later, when data mining and knowledge discovery advanced, and constituted their own separate scientific discipline, this also contributed greatly to the development of clustering methods. The special focus there has been on the computationally efficient algorithms for large data sets [167, 170, 369, 96]. Perhaps due to the interdisciplinary nature of the cluster analysis, the same methods are often invented with different names on different disciplines.

There exist a huge number of clustering applications from many different fields, such as biological sciences, life sciences, medical sciences, behavioral, and social sciences, earth sciences, engineering and information, policy and decision sciences, to mention just a few [9, 204, 220, 203, 106, 400]. This emphasizes the importance of data clustering as a key technique of data mining and knowledge discovery [170, 167, 96, 149, 30, 142, 369], pattern recognition [372, 93, 94, 205, 135], and statistics [87, 175].

The range of clustering applications is very wide. It includes analysis of software modules and procedures [266, 421, 422, 399], grouping customers of similar behavior in marketing research [31], classification of unknown radar emitters from received radar pulse samples [259], optimal placement of radioports in cellular networks [2], identification of subtypes of schizophrenia [186], archeological applications [7], peace science applications (identification of international conflicts [397]), P2P-networks [327], computer vision [132], etc. The list above is almost endless. It also contains some quite exotic examples.

What is a cluster?

"Do not forget that clusters are, in large part, on the eye of the beholder."[99]

As one can see from Figure 5, the recognition of clusters in a two-dimensional

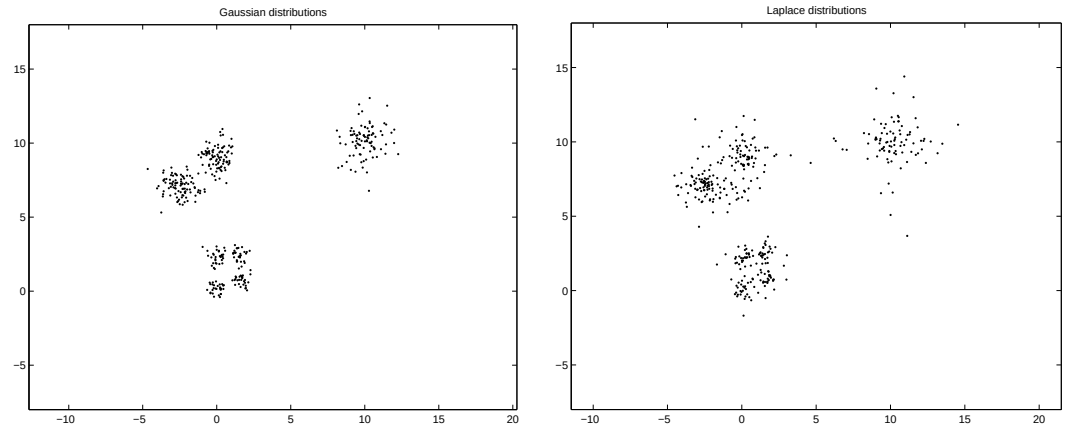


FIGURE 5 Ambiguous cluster models. On the left, data is drawn from a mixture of the normal distribution, and, on the right, from a mixture of the Laplace distribution.

view is a relatively easy problem. But, it is not as simple to tell the number of clusters for the depicted data sets. Especially, the Laplace-case on the right is problematic in this sense, because the clusters are more spread in all directions. The number of clusters is ambiguous, because the clusters are separable on at least two hierarchical levels. One may find either 3 or 7 clusters in the both data sets.

Although clusters are often visually observable in low-dimensional spaces, it is no easy matter to give a formal definition for a cluster. The definitions tend to be weak and dependent on applications [369]. Clusters may be of various shape and size, or influenced by the goal of the cluster analysis in a special way. For instance, one analyst may need a more detailed view to the data than the other. This is illustrated in Figure 5, which shows that the number of inherent clusters may change according to the resolution (local versus global) with which one is looking at the data [204, 369]. Overall, the lack of a universal definition for the term cluster is frequently addressed by researchers in the cluster analysis literature, and they tend to agree that giving that definition is an intractable problem [369, 400, 106, 220, 8, 80].

Clustering methods often tend to yield a data description in terms of clusters that have strong internal similarities or, in other words, that contain points which are close to each other in some sense [93]. Hence, sometimes the internal properties are not enough and the cluster is defined in terms of internal cohesion (homogeneity) and external isolation (separation). This means that a cluster is considered as a collection of objects that are similar to one another within the same cluster and dissimilar to the objects that are located in other clusters [167]. From this, an interesting connection to the field of software engineering can be recognized, since the principle is very similar to the common software architecture rule, which is based on the principle of "*loose coupling and strong cohesion*". Such an architecture aims to localize the effects caused by code modifications (see, e.g., Bachmann et al. [15]). Software components with a large number of

links from one to another can be considered "similar" or "close" to each other. A good software architecture contains usually clearly separated "component clusters".

Next, some common definitions, that are known from the literature, are given for cluster [204, 8].

- "A cluster is a set of entities which are alike, and entities from different clusters are not alike."
- "A cluster is an aggregation of points in the space such that the *distance* between two points in the cluster is less than the distance between any point in the cluster and any point not in it."
- "Clusters may be described as connected regions of a multidimensional space containing a relatively *high density* of points, separated from other such regions by a region containing a relatively low density of points."

Although cluster is an application dependent concept, all clusters possess certain properties: density, variance, dimension, shape, and separation [8]. The clusters are expected to be tighter and more compact high-density regions of data points when compared to the rest of the problem space. From compactness and tightness, it follows that the within-cluster dispersion (variance) is relatively small. The shape of a cluster is not known a priori. It is determined by the selected algorithm and clustering criteria. Separation is defined by the degree of cluster overlapping and the distances of clusters to each other. Fuzzy clustering methods produce overlapping clusters by assigning the degree of cluster membership for each point to all clusters [21, 22]. Traditional partitioning clustering methods, such as K-Means, and hierarchical methods produce distinct clusters, which means that each data point is assigned to only one cluster. The dimension of a cluster is defined by the space of the problem variables. For spherical shaped clusters, it is possible to calculate radius. These are the measurable features for any cluster, but it is not possible to assign universal values or relations to them. Perhaps the most problematic features of clusters are their shape and size.

3.2 Elements of clustering process

Although the intuitive idea of cluster analysis is simple, the successful completion of a clustering task on real data requires a large number of correct decisions and choices between several alternatives. According to Anderberg [9], cluster analysis consists of nine major elements. Because real-world data sets are often incomplete, that is, some values are missing, the list is extended with the item of missing data strategy [258, 208]. Data presentation is another important element due to the large number of different algorithms. When the choice between different algorithms is made, one should take into account the requirements for the

data presentation. While some clustering algorithms operate directly on data values, others are based on (dis)similarity matrices. For example, prototype-based clustering methods are useless if only between-object (dis)similarities are available and the absolute values of the object attributes are unknown. On the other hand, size of similarity matrices explodes on large data sets, which makes the methods based on them useless in those cases. The following list includes the most significant elements of the general clustering process.

1. Data presentation.
2. Choice of objects.
3. Choice of variables.
4. What to cluster: objects or variables.
5. Normalization/weighting of variables.
6. Choice of (dis)similarity measures.
7. Choice of clustering criterion (objective function).
8. Choice of missing data strategy.
9. Algorithms and computer implementation (and their reliability, e.g., convergence).
10. Number of clusters.
11. Interpretation of results.

Jain et al. [205] suppose that data collection, data representation, normalization, and cluster validity are as important factors for the successful clustering process as the clustering strategy itself. Hastie et al. [175, p.459] suggest that the choice of (dis)similarity measures may be even more important than the choice of the clustering algorithm. The list could also be extended with cluster validation [204, 8], but this is not seen as necessary as it closely relates the estimation of the correct number of clusters to the result interpretation. Visual exploration of the obtained clusters is a kind of validation technique.

One should note that use of any mathematical validity index risks the philosophical nature of clustering problems. As it is demonstrated in this chapter (cf. Figure 5), there does not always exist a unique or correct clustering model. Hence, visual interpretation and validation by domain experts remain important approaches to the determination of the correct number of clusters, or, in fact, even to the selection of clustering criteria. In the following, a more detailed treatment for the elements is given.

3.2.1 Data representation

The target of cluster analysis can be a data sample drawn from a particular population by using some statistical data collection strategy. On the other hand, in DM applications the target data is often collected without statistical data collection strategies [170]. A statistical data sample can represent, for example, TV-viewers or customers of a supermarket. A more arbitrarily collected data set may consist of values of numerous setting and measurement parameters that are stored, for example, from an industrial process in the long run. Both of these types of data can be presented in an $n \times p$ *data* or *pattern matrix* [204]:

$$\mathbf{X} = \begin{pmatrix} x_{11} & \dots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \dots & x_{np} \end{pmatrix}.$$

The rows of the matrix correspond to the data objects and the columns to the variables. Depending on the context, a data object might be an observation, a pattern, an instance, an event, a data unit, a tuple, or a case. From mathematical and computational point of view, each row is equivalent to the transpose of $p \times 1$ data vector, which represents a point in a p-dimensional space (or, in the case of missing values, in its subspace).

For example, the hierarchical single-link clustering method is based on $n \times n$ *(dis)similarity* or *distance matrix*

$$\mathbf{D} = \begin{pmatrix} d_{11} & \dots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{n1} & \dots & d_{nn} \end{pmatrix},$$

which represents pair-wise proximities between objects in a data set. If not otherwise announced, the term dissimilarity matrix is used from now on. Clearly, the size of matrix $\mathbf{D}$ is n^2 and $d_{ij} = 0$ for all $i = j$. d_{ij} can be any l_q-norm, dissimilarity measure for categorical data, etc. Distance and (dis-)similarity measures will be considered more precisely later in this thesis. The problem with this data presentation approach, especially on large data sets, is n^2 memory requirement. For example, on a 32-bit computer architecture, a double-precision dissimilarity matrix of 5000 data objects takes 200 megabytes of memory. This introduces demands for efficient memory utilization. Another risk might be the missing data values. The similarities computed in different sub-spaces are not easy to compare. Sometimes the comparison may prove impossible. For instance, let us consider the distance computation for the following three 3-dimensional data vectors

$$\mathbf{x}_1 = (1 \quad 0 \quad \text{NaN})^T, \quad \mathbf{x}_2 = (\text{NaN} \quad 1 \quad 1)^T, \quad \mathbf{x}_3 = (1 \quad \text{NaN} \quad 1)^T.$$

Using any method, straightforward comparison of the between-object distances is difficult, since all the points lie in the different sub-spaces. Imputation could be a solution, but it is not a sensible operation when nothing is assumed about data a priori. Missing data strategies are considered more precisely later in this thesis.

3.2.2 Data collection strategy

The choice of data objects, also known as a data collection strategy, is an important issue from a statistical perspective. In DM and KD, all data is usually employed, which makes data collection strategies less significant when compared to statistical analysis. Hence, in data mining one rather speaks about secondary data analysis where the data already exists for processing [170]. Therefore, in a data mining problem, the data set may represent the whole population which one wants to compress, summarize, and explore in order to find novel and useful structures and, thereby, to understand it better.

3.2.3 Feature selection and extraction

The choice of the operative variables is known as *variable* or *feature selection*. The goal is to find the most effective subset of features before the actual cluster analysis phase [203]. An alternative for the feature selection is *feature extraction*. For finding the most discriminative variables of a given data set, one reduces the dimension and, at the same time, tries to preserve the discriminative power of the data set. If the value of a variable is approximately constant for all the objects, the discriminative power of that variable is very low. Such a variable is likely to be unnecessary in cluster analysis. Feature extraction can be realized by projection methods, such as MDS, PCA, or independent component analysis (ICA) (see methods, e.g., in [175, 200]). These methods reduce the number of variables by projecting the data to new coordinate axes that represent the most informative directions (latent variables) of the data. The new variables are linear or non-linear transformations of the original variables [203, 170]. These methods can be used either separately or in sequence. Clustering methods can be used for feature selection as well. By taking appropriate (dis-)similarity measures for variables, such as correlation coefficients, alike variables can be grouped and presented by one representative variable[1]. On the other hand, variable selection is closely related to data scaling or weighting, which is discussed more closely in the context of data standardization in Section 3.2.5. The rejection of a variable corresponds with the scaling of a variable to a zero-length interval or weighting by zero [145].

3.2.4 What to cluster?

A starting point of all cluster analysis tasks is the data itself whose internal structure is the matter of interest. The characteristics of a data set, such as noise, gross errors, incompleteness or sparsity, and the amount of data itself, compose the requirements and constraints for the problem induction and, thereby, for the selection of clustering algorithm. Therefore, careful analysis of the target data set together with prior knowledge often precedes the problem formulation and algorithm selection. One may want to cluster either data objects (row-wise) or vari-

[1] For instance, Clustan Software (www.clustan.com) and MATLAB (www.matlab.com) offer methods for variable clustering.

ables (column-wise), or sometimes even both (c.f., Bailey's definitions in Section 3.1). From this arises the need for several different similarity measures. Usually the similarity of objects is defined through the distance (e.g., l_q-norms) between two data points, whereas in the case of variables, correlation is perhaps the more appropriate measure (see Section 3.2.6). The choice of clustering units, either objects or variables, is an application dependent decision.

Data types

In typical DM applications, it is not usually possible to inspect variables individually. However, it may be useful to be aware of basic properties of the variables, since this information can be utilized in the preprocessing and algorithm selection. Data types are usually categorized into three main classes: *binary*, *discrete*, and *continuous* [9, 204, 220]. A data type refers to the degree of quantization in the data.

Continuous data types, such as height or weight of a person, can take any real value in a fixed range of values. This is perhaps the most frequently encountered data type. Continuous variables are of two scales: interval and ratio.

Binary and discrete variables are categorical data types. A discrete variable has a finite number of possible states, such as colors (red, blue, green). The states may have mutual ordering or not. Clustering algorithms for categorical data sets are proposed, e.g., in [192, 151, 184]. Categorical variables are typical, for instance, for market basket analysis [151]. Discretization or quantization of continuous real-world analog signals is usual in a number of modern signal processing systems. Hence, there is a large number of systems that produce categorical (digital) data. Applications of this kind can be found in several fields such as audio/video signal processing, speech processing/recognition, digital image processing, digital communication, industrial process control and analysis, etc.

Binary variables are the simplest form of data. Binary variables have exactly two values, for example, $0-1$,"yes-no","male-female" or "smoker-nonsmoker". Categorical variables can be represented using binary attributes (for example, use values 0 or 1 to denote the similarity of categorical observations). Other advantages of the binary data are that they are noise-free and on the same scale in all dimensions [307]. The binary variables are divided into two classes, namely, *symmetric* or *asymmetric*. Symmetry of a binary variable depends on the relative significance between two alternative states. If both states are equally important, the binary variable is symmetric. In the case of an asymmetric binary variable, either of the states is more informative than the other. Hence, for symmetric binary variables, *negative* (zero-zero) *matches* ($a = 0$ and $b = 0$) are weighted equally with *positive* (one-one) *matches* ($a = 1$ and $b = 1$). In the asymmetric case, the negative and positive matches are not equally important. The positive matches are often used to denote the more significant similarity [106]. The more significant one of the states represents more aberrant phenomena and the other represents more usual facts. An asymmetric binary variable may represent, for example, occurrence of some infrequent sickness, such as AIDS. MONA, a monothetic di-

visive hierarchical clustering algorithm is intended especially for binary data, see Kaufman et al. [220, Ch.7]. Ordinez [307] proposes a specialized variant of the K-means-type algorithm for binary data.

Data scales

Data features can also be categorized according to scale [204]. There are two principal data scales: *qualitative* or *quantitative*.

A qualitative scale is either *nominal* or *ordinal*. The nominal scale generalizes the binary data type. The numbers are meaningless in the quantitative sense and they are used as labels. The four state nominal variable expressing the nationality of a person using numerical coding serves as a good example of the nominal data types: 1=Finnish, 2=Swedish, 3=Norwegian, and 4=English. In this case, the numbers provide proper labels for computer operations, but no ranking for the nationalities. A clustering algorithm for nominal variables is proposed, e.g., by Ganti et al. [137].

Another qualitative scale, namely ordinal, is the weakest numerical scale with meaningful numbers. It possesses a finite set of states that comprise a meaningful sequence of the codes (e.g., $1, 2, ..., M$) with an order. The farther apart the two codes are from each other in a sequence, the greater the distance between them. Therefore, the relations of the numbers have a meaning, but the mutual ratios are meaningless. For example, the following numberings $(1, 2, 3)$, $(10, 20, 30)$ or $(1, 20, 300)$ have an equivalent meaning on the ordinal scale, since the first and the last elements are equally distant to each other in each case. A simple example of an ordinal variable is the order of competitors in a race. Although the exact results (e.g., running time, length of jump, etc.) for each competitor might be unavailable, it is possible to express, using competitor rankings, that the winner and the last (M^{th}) competitor are the most apart from each other. It is just left unknown how far apart from each other they are according to some measured quality. Another typical ordinal scale application is, for instance, questionnaire data that model, for example, appreciation of a thing (food, sport, painting, song, etc.) by using different grades, such as 1 = detest, 2 = dislike, 3 = indifferent, 4 = like, 5 = admire. Hence, the order of the numbers is again meaningful. The higher the number the more pleasing the thing. Because of the lack of knowledge about absolute differences between the states of discrete variables, the order-statistics that are based on l_1-norm are often employed in the analysis.

Interval and *ratio scales* are quantitative by nature. On the interval scale, a unit of measurement exists and the interpretation of the numbers depends on this unit. Hence, numbers on the interval scale can not be interpreted before the actual scale is known. Temperature units, Fahrenheit and Celsius, are examples of the interval scaled variables. Both of them are continuous, but their ratios of numbers have different meanings.

The ratio scale is the strongest scale and in this scale the numbers have absolute meaning. Hence, zero has the same meaning for all measurement units. For instance, distance units (centimeters, meters, miles, etc.) and monetary units

(Euro, US dollar, the pound, etc.) are ratio scaled variables, since multiplying a value by a given factor has always the same significance, no matter what unit is used.

Mixed variable data

In real-life, it often happens that more than one data type occurs in the same data set. In these cases we are dealing with mixed data types, which offer new challenges to the distance calculation and algorithm development. Clustering algorithms for mixed data types are given, e.g., in [70, 192, 183]. Podani [321] introduces three different approaches to deal with mixed variable data sets (see also details in [9]). The first one of these is the scaling of variables to the same scale. The problem is that this may lead to a loss of information, if the conversion happens from continuous towards some unordered nominal scale. On the other hand, scaling from an unordered discrete scale towards continuous ordered scales requires some domain knowledge to link the unordered classes with an external ordering. The second approach is to separately analyze each variable by taking into account the data type and to synthesize the results. The third approach is to use specific coefficients for data types. All these classic approaches give a burdensome impression when considered from the DM point of view. When one is dealing with data that consists of hundreds or thousands of variables - and these are not extraordinary numbers with DM applications - the methods that require variable-wise external knowledge are no more practical. Hence, robust and automated methods are needed for handling large data sets with various data types and scales.

3.2.5 Standardization

Standardization of variables equalizes the weights of different variables during the cluster analysis process. This is necessary, since in many applications different variables are measured in different units, which leads to unequal scales and, thereby, unequal contributions in the clustering process. For example, let us consider distance computation of bivariate data representing weight and age of a person. If we take milligrams and years as units and compute the Euclidean distance, it follows that the variable "weight" will have a dominating effect to the result. Considerable differences in the range of the variables may hinder their simultaneous visual interpretation as well [339, p.60].

A typical form of standardization, also known as the "z-score" formula, is to transform a variable $x_i = \{(\mathbf{x}_j)_i, \quad j = 1, \ldots, n\}$ to zero mean and unit variance [106, 284]. Hence, the "z-score" transformation of a scalar variable $x_i \in \mathbb{R}$ is realized by

$$\hat{x_i} = \frac{x_i - \mu}{\sigma} = \frac{1}{\sigma} x_i - \frac{\mu}{\sigma} = \alpha x_i - \beta, \tag{1}$$

where $\hat{x_i}$ is the scaled variable, μ the sample mean, and σ the standard deviation. From the right-hand side of the formula one can see that this is actually only a linear transformation of a variable. Hence, as pointed out by Saalasti [339, p.60],

by determining the coefficients α and β in a different way, other approaches, such as division by the range, which scales a variable into a given interval (e.g., $[0,1]$ or $[-1,1]$), are obtained. A variable is scaled to unit range $[0,1]$ by choosing

$$\alpha = \frac{1}{\max\limits_i x_i - \min\limits_i x_i}$$

and

$$\beta = -\frac{\min\limits_i x_i}{\max\limits_i x_i - \min\limits_i x_i}.$$

Milligan et al. [284] present a simulation study for seven standardization methods. In several cases, the division by the range of a variable produces the best recovery for the underlying cluster structure. It is also robust across wide variety of conditions. The traditional unit variance scaling proved to be less effective than the range scaling.

A potential problem for the unit variance and range scaling methods is the existence of outliers. The classical statistics used in the unit variance scaling are sensitive to any gross errors. In the range scaling approach, one gross error forces the rest of the data to the other end of the range. Although being robust against outliers, rank-based standardization, in which the values are simply converted to ranks, has very poor performance under most conditions in [284].

As the examples in [106, p.49] show, the standardization of variables with respect to the standard deviations over a complete set of objects does not always work in the case of cluster analysis. In some cases, it may reduce the influence of the most discriminative variables due to the decreased weights of the variables with large between-group variations. Therefore, in cluster analysis, it may be more sensible to prioritize within-group variations in standardization of cluster data. However, the problem is that the groups are not known a priori. To solve this problem, e.g., Huang et al. [190] propose a W-K-means algorithm that iteratively updates the weights with respect to a current partition. Many references to other methods are found therein. When unit variance or range scaling is applied to the within-cluster scaling, one should note that shifting the mean of the clusters to zero or transforming the range of the cluster data to a given range leads to overlapping clusters. Therefore, the inter-cluster scales must also be taken into account in standardization [284].

Other problematic issues are standardization of online clustering data sets and categorical data types. In the first case, the sample range of online data may be difficult to predict. In the latter case the data type may represent an unordered scale. Another problem that is related to categorical data types is the transformation of variables into a continuous interval. When all variables in a data set, both discrete and continuous, are forced to the continuous range from zero to one, it is not clear in the case of binary variables whether the scaled values should be zero and one (end-points of the range) or the mid-points of the both halves (0.25 and 0.75). The first scaling provides a maximal weight for each binary variable and,

thereby, also maximal discriminative power. Hence, the binary variables contribute to the distance computation with maximal influence. The latter scaling would average the effect, which means that two values would not be maximally distant to each other on the new scale, but would be representative values for the two halves of the variable range.

Actually, the same problem generalizes to all categorical data types, but fortunately the significance decreases as the number of states of the categorical variable grows. One should also note that when the chosen method is scale invariant (e.g., by relying on marginal distributions [23]), there is no need for standardization procedures. For example, minimum, maximum, mid-range, or the coordinate-wise median are order-statistics, which are inherently scale invariant.

3.2.6 Choice of proximity measure

Cluster analysis is, for a large part, based on comparison of how similar or dissimilar data objects are to one another. Similar objects are gathered into the same group whereas dissimilar objects belong to distinct groups. Data objects or observations are described by a set of variables (features, attributes, etc.), which actually represent many kinds of real-world observations and measurements. By analyzing the type of these variables (binary, discrete, continuous, etc.), one can choose the most appropriate distance measure. There are several terms that are used in this context of proximity measures, e.g., *similarity*, *dissimilarity*, *distance* or in more general terms, *proximity*. In this thesis, the terms distance and similarity are mostly used. A large distance between two objects corresponds to a small similarity and vice versa.

A comprehensive introduction to similarity measurements, for both variables and objects, is given by Anderberg [9]. Shorter presentations are given, e.g., in [106, 220, 204]. Properties of (dis)similarity measures for binary data are discussed by Zhang et al. [415]. An examination of eight binary vector dissimilarity measures in handwriting identification task is presented by Zhang et al. [414]. In the following, some of the most common classes of distance functions will be presented.

l_q-norms

A general class of distance functions in p-dimensional vector space, satisfying the conditions of metric spaces (see, e.g., [297, 66]), is defined by l_q-norm (a.k.a. *Minkowski metric*) [66, 94, 412]:

$$l_q(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{p} |(\mathbf{x})_i - (\mathbf{y})_i|^q \right)^{1/q} = \|\mathbf{x} - \mathbf{y}\|_q, \quad q < \infty, \tag{2}$$

where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$.

The most common special cases of (2) are l_1-, l_2- and l_∞-norms. l_1-norm (a.k.a *Manhattan* or *City-block distance*) is equal to the sum of the shortest projec-

tions parallel to coordinate axes:

$$l_1(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{p} |(\mathbf{x})_i - (\mathbf{y})_i| = \|\mathbf{x} - \mathbf{y}\|_1.$$

l_1-norm is related to discrete variables. It is also the norm behind the coordinate-wise sample median estimate [213]. The popular l_2-norm (a.k.a. *Euclidean distance*) is defined by:

$$l_2(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{p} |(\mathbf{x})_i - (\mathbf{y})_i|^2 \right)^{1/2} = \|\mathbf{x} - \mathbf{y}\|_2.$$

l_∞-norm is the maximum distance in the direction of any coordinate axes:

$$l_\infty(\mathbf{x}, \mathbf{y}) = \arg\max_{1 \le i \le p} |(\mathbf{x})_i - (\mathbf{y})_i| = \|\mathbf{x} - \mathbf{y}\|_\infty.$$

Mahalanobis distance

The *Mahalanobis distance* is a data-dependent metric that takes correlations between variables into account [94, 170, 106]. If there is strong correlation between two or more variables, the weights of these variables will be cut down by Mahalanobis distance. This may occur when some measurement is repeated several times for each object or observation and it becomes more significant than the others in non-standardized distance computation. The squared Mahalanobis distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ reads as:

$$d_{mh}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \mathbf{y}), \tag{3}$$

where $\boldsymbol{\Sigma}$ is the sample covariance matrix. The sample (co-)variance matrix incorporates the correlation between variables and standardizes each variable to zero mean and unit variance. When correlation between variables is zero and each variable has unit variance ($\boldsymbol{\Sigma}$ is the identity matrix), the Mahalanobis distance equals to the squared Euclidean distance.

Correlation coefficients

The *Pearson's product-moment correlation coefficient* (also referred to simply as *coefficient of correlation* or *Pearson's correlation*) is a similarity index for ratio and interval scaled variables [357, 204]. Given a set of n paired measurements or observations (x_i, y_i), it is computed as

$$r_{xy} = \frac{s_{xy}}{\sqrt{s_{xx} s_{yy}}} \tag{4}$$

where $s_{xy} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$, $s_{xx} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2$ and $s_{yy} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y})^2$. $\bar{x}$ and $\bar{y}$ are the sample means of x_i and y_i ($i = 1, \ldots, n$), respectively. The absolute value $|r_{xy}|$ can be used when negative and positive correlation have the

same significance [204, p.16]. The corresponding dissimilarity index on the interval $[0,1]$ is obtained by $d_{xy} = 1 - |r_{xy}|$. The correlation coefficients can not be used to measure the magnitude of the difference between two objects or variables. Despite of being utilized also as an index of inter-object similarity (see, e.g., [8, p.22-24]), the correlation coefficients are more often used to measure linear dependency between two variables. Hence, this is a potential similarity index for variable grouping by clustering methods. Note that Pearson's product-moment correlation coefficient is based on the classical parametric statistics and normal distribution, which makes it sensitive to distortions. Robust variants of the correlation coefficients are non-parametric, and distribution-free rank-based methods, such as Spearman rank correlation coefficient and Kendall rank correlation coefficient, may be useful when the sample is on ordinal scale or distorted and not normally distributed (see details, e.g., [357, p.231]).

Cosine similarity

The *cosine similarity* gives the cosine of the angle between two vectors. It is a popular index of similarity in text clustering [345] defined as:

$$d_c(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \mathbf{y}}{\|\mathbf{x}\|_2 \cdot \|\mathbf{y}\|_2}. \tag{5}$$

The cosine similarity is scale invariant and independent on the length of the vectors. Hence, it considers the proportions of feature vectors, but ignores their total lengths. This is useful, for instance, in comparison of a pair of text documents with the same proportions, but with a different number of term instances, since such documents are usually considered similar to each other. The cosine similarity presumes orthogonality of the features in vector space, which does not hold for document collections if the documents have something in common and are not completely independent [345]. Because the cosine similarity index does not satisfy the triangle inequality axiom, it is not a metric [363].

General similarity measure for missing data

Kaufman et al. [220] present a general distance measure in the context of the large-scale clustering algorithm CLARA. The distance measure resembles the *Gower's general similarity measure* that was initially intended for data sets that contain both continuous and categorical variables (see [147] or [106, pp.43-44]). The Euclidean distance between two p-dimensional real-value vectors $\mathbf{x}$ and $\mathbf{y}$ is then defined as

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\frac{p}{\hat{p}} \sum_{i=1}^{p} ((\mathbf{x})_i - (\mathbf{y})_i)^2}, \tag{6}$$

where $\hat{p}$ is the number of variables that exist in both $\mathbf{x}$ and $\mathbf{y}$.

Distance measures for binary variables

A large number of similarity and distance measures for binary data are proposed in the clustering literature [9, 106]. A very recent and thorough review is given by Cha et al. in [61]. As a special case of categorical data, the transformation to the continuous scale is not necessarily trivial. All continuous distance measures are not necessarily reasonable in binary data cases, but many binary counterparts are presented for clustering applications [9].

One of the most popular distance measures for comparing two binary vectors is the *Hamming distance*, which is defined as the count of "bits" that differ in the two vectors [61]

$$d_{hamming} = \sum_{i=1}^{p} \|(\mathbf{x})_i - (\mathbf{y})_i\|_1.$$

One can easily see that the Hamming distance is actually equal to the aforementioned l_1-distance in the binary vectors case. The Hamming similarity is defined as

$$S_{hamming} = \sum_{i=1}^{p} s_i, \quad \text{where} \quad s_i = \begin{cases} 1, & \text{if} \quad (\mathbf{x})_i = (\mathbf{y})_i, \\ 0, & \text{otherwise.} \end{cases}$$

There are also many binary similarity measures that are derived from the inner product of two vectors as

$$S_{innerproduct}(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y}. \tag{7}$$

The counts of matches between a pair of binary vectors are usually presented by a *contingency table* (Table 1). The table contains the number of matches and mismatches between two p-dimensional vectors [220, 106, 167]. Perhaps the

TABLE 1 A binary contingency table.

	1	0	sum
1	n_{11}	n_{10}	$n_{11} + n_{10}$
0	n_{01}	n_{00}	$n_{01} + n_{00}$
sum	$n_{11} + n_{01}$	$n_{10} + n_{00}$	$p = n_{11} + n_{10} + n_{01} + n_{00}$

most difficult issue with the binary measures is whether to include or exclude negative matches (case n_{00} in Table 1). As mentioned in Section 3.2.4, binary data types are of two types: symmetric or asymmetric. In the case of an asymmetric binary variable, the positive match provides more information about the objects than the negative match and, thereby, they are weighted more in distance computation. Some measures completely ignore the negative matches.

A large number of common binary similarity measures are derived from the following generic formula:

$$S(\mathbf{x}, \mathbf{y}) = \frac{n_{11} + \beta n_{00}}{n_{11} + \beta n_{00} + \lambda(n_{10} + n_{01})}. \tag{8}$$

The influence of the co-absence ("$0-0$") features is adjusted by the choice of parameter β. Correspondingly, the influence of mismatches is adjusted by changing the value of parameter λ. More information about binary measures derived from (8) can be found, e.g., in [61, 414, 415, 400, 9, 106].

3.2.7 Choice of clustering criterion

Optimization-based clustering algorithms, such as K-means, partition a data set by minimizing or maximizing some numerical criterion. Hence, the clustering criterion defines what kind of partition one is looking for. It is a very difficult problem to formulate a criterion function that exactly corresponds to the quality of classification in real-world problems [29]. A variety of clustering criteria derived from the dissimilarity matrix and directly from continuous variables are suggested, e.g., in [106, 94]. The most commonly used criterion is the sum of the within-group sums of squared distances over all variables, which is equal to minimizing $\mathrm{tr}(\mathbf{W})$, where $\mathbf{W}$ is the within-group dispersion matrix. The trace of a matrix is the sum of the diagonal elements. This equals to minimizing the sum of the squared Euclidean distances between data points and their cluster means, which corresponds to the criterion that the well-known K-means-algorithm minimizes. Weaknesses of $\mathrm{tr}(\mathbf{W})$ criterion are the scale-dependency and the inherent assumption about the spherical shape of clusters, even though the natural clusters can be of other shapes. In order to avoid the problems of the $\mathrm{tr}(\mathbf{W})$ criterion, several alternative criteria are suggested. For example, the $\det(\mathbf{W})$ criterion allows elliptical shape of clusters. This, however, assumes that all clusters have the same orientation and elliptical degree. The scale-dependency is avoided by maximizing $\det(\mathbf{T})/\det(\mathbf{W})$ or $\mathrm{tr}(\mathbf{B}\mathbf{W}^{-1})$, where $\mathbf{T}$ is the total dispersion matrix of data $\mathbf{X}$ and $\mathbf{B}$ the between-group dispersion matrix (for definitions, see [106, p.92–93]). Other criteria for clusters of different shapes and sizes are presented, e.g., in [106].

3.2.8 Missing data

"*Incomplete, noisy, and inconsistent data are commonplace properties of large real-world databases and data warehouses*" [167, p.106]. Besides that real-world data sets are often large, many times even huge, they are often incomplete and noisy as well. An incomplete data set contains objects, in which one or more features are missing. In order to minimize the bias in the data, missing data values must be treated carefully. There are many reasons for the absence of data values, such as technical malfunctions, death of patients, refusal of respondents to answer certain questions, etc. [27]. The underlying reason defines the distribution for the missing data. Knowledge about the statistical characteristics of the missing data source helps one to choose the best strategy to deal with it. One should also note that missing data encompasses information about the data set. Pyle [324] emphasizes that missing data patterns may sometimes contain the most important piece of information of the target domain. In such cases, the information about missing

data should not be completely lost in DM operations. Therefore, it may be wise to capture the information about missing data patterns before substituting them with new values.

Missing data mechanisms

Roughly speaking, there are basically two types of missing data values [324, 258]. The first type consists of values that are, for one reason or another, not entered into the data set, although the true underlying values exist. There are many possible reasons for the lack of these values, for example, a human mistake or a fault in the data gathering or storage system. In some cases, empty values of a data set correspond with measurements for which there exist no value in the real world. For example, it is not possible for an unemployed person to have an employer. Also, some individuals questioned may not have an opinion. Hence, information really does not exist for these fields. On the other hand, this type of "do not exist" or "do not know" observations are actually information by themselves. For example, the most important target group for the presidential election campaign are the people who are uncertain and do not have an opinion yet.

In statistics, missing data values are divided into three classes depending on the mechanism responsible for the absence of data [258, 141, 27]:

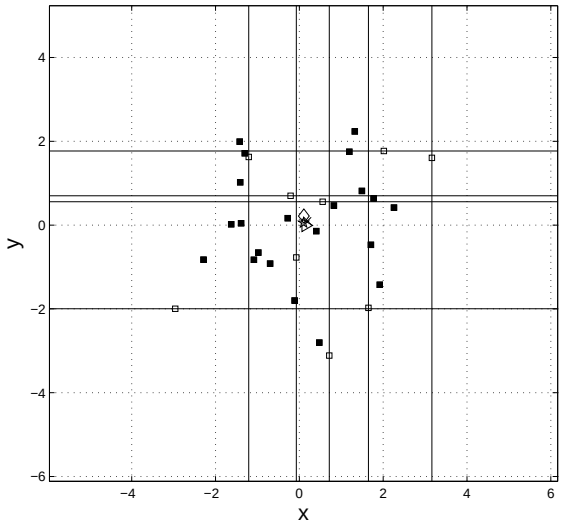


FIGURE 6 A data set with 15% of missing values that are generated completely at random (MCAR).

Missing completely at random (MCAR). The probability that the j^{th} component of a vector $\mathbf{x}_i \in \mathbb{R}^p$ $(i = 1, \ldots, n)$ is missing is independent of any other known or missing values of $\mathbf{x}_i$. In MCAR case any missing data technique can be used without producing significant bias on the data [27]. This is illustrated in Figure 6, where 15% of data is removed completely at random. The missing values are presented by lines, which mean that any value is possible. The sample mean and the spatial median of the data set before the data deletion are marked by '$\times$' and '$\star$', respectively. One can see that the

sample mean ('▷') and the spatial median ('◇') are almost unbiased after the deletion of the points, when the available case strategy (see Section 3.2.8) is applied to the computation.

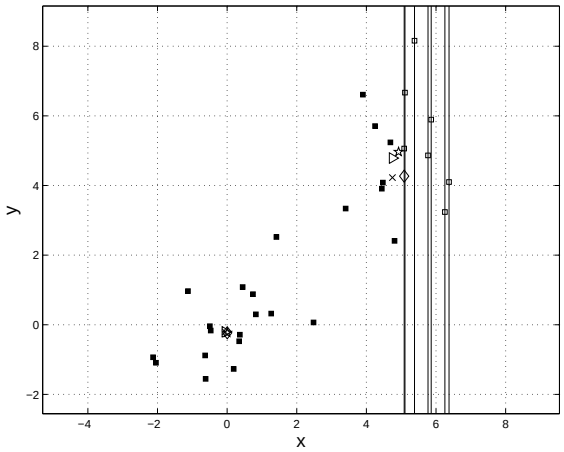


FIGURE 7 A data set with some values missing at random (MAR). If $x > 5$ then y is missing.)

Missing at random (MAR). The probability that the j^{th} component of a vector $\mathbf{x}_i \in \mathbb{R}^p$ is missing does not depend on the values of missing components, but may depend on the values of observed components. Hence, the observed values of $(\mathbf{x})_j$'s is a random sample of the sampled values within subclasses defined by the values of $(\mathbf{x}_i)_k$ where $k \neq j$, and provided that $(\mathbf{x}_i)_k$ is always available. Figure 7 shows two cluster data, where the value of y variable is missing for all $x > 5$. Hence, the missing values exist only in one cluster. They are marked by the lines. The sample means and the spatial medians of the clusters before the data deletion are marked by '$\times$' and '$\star$', respectively. One can see that the sample mean ('▷') and the spatial median ('◇') are almost unbiased after the deletion of the points, when the available case strategy is applied to the computation. However, the variation of the estimates in the cluster with incomplete data is larger than the estimates of the complete cluster.

Not missing at random (NMAR). The probability that the j^{th} component of a vector $\mathbf{x}_i \in \mathbb{R}^p$ is missing depends on its value. Hence, the availability of $(\mathbf{x}_i)_j$ is a function of itself. For example, if a value of some measurement is out of some permissible range, it might be therefore censored and replaced by an empty value. In this case, we have a partial information that the value has exceeded some range limits and the mechanism of missing data is understood. However, no general method exists for this kind of mechanism and standard complete case analysis with such data are in general biased [258, 141]. In Figure 8, one observes that the estimates are highly biased on this incomplete data set. The sample means ('$\times$','▷') and spatial medians ('$\star$','◇') are computed for complete and incomplete data, respectively, using the available case strategy.

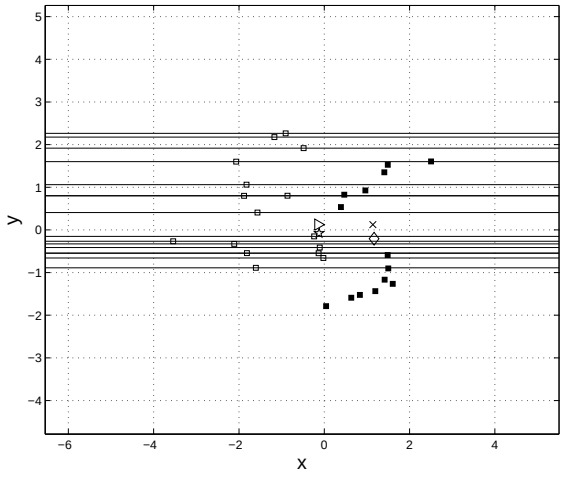


FIGURE 8 A data set with values not missing at random (NMAR). Variable x exist only if $x > 0$.

Figures 6–8 show that missing values are kind of outliers, since the true values can be actually any permitted value. This is illustrated by depicting the objects containing one or more missing values by a straight line (in principle the line goes from $-\infty$ to ∞). Hence, as well as outliers, the missing data values are another relevant justification for utilization of robust procedures in data analysis tasks (e.g., cluster analysis and/or data mining).

Strategies for handling missing data

In order to deal with different types of incomplete data sets, a treatment for missing data values must be implemented into clustering algorithms. Little and Rubin [258] divide the missing data methods into the following, not mutually exclusive, categories: 1. Procedures based on completely recorded units, 2. Imputation-based procedures, 3. Weighting procedures, and 4. Model-based procedures. Most of the methods are able to deal with MCAR data, but MAR and NMAR are more complicated from the statistical perspective. In this thesis, it is assumed that in DM applications data is missing completely at random. In the following, some simple methods will shortly be presented [27, 106, 258].

Complete case strategy This is the most straightforward strategy for handling missing data. Complete case methods simply utilize only complete cases from the target data. After the incomplete objects have been pruned away, it is possible to apply ordinary clustering algorithms as if the data were complete. The weakness of this approach is the risk of losing too much of data, which may lead to biased estimates. Figure 9 presents an example. Incomplete samples $\mathbf{x}_2$ and $\mathbf{x}_3$ are ignored before the mean is computed. The complete case handling leads to 40% loss of data before the averaging.

Strategy of using available data Methods based on available data strategy utilize all available data in computation. If the j^{th} value of a p-dimensional object

$$\mathbf{X} = \begin{pmatrix} 1.1 & 2.7 \\ \text{NaN} & 5.7 \\ 1.1 & \text{NaN} \\ 0.6 & 2.4 \\ 1.9 & 0.9 \end{pmatrix} \quad \Rightarrow \quad \boldsymbol{\mu} = \frac{1}{3}(\mathbf{x}_1 + \mathbf{x}_4 + \mathbf{x}_5) = \begin{pmatrix} 1.2 \\ 2.0 \end{pmatrix}$$

FIGURE 9 Complete case strategy: an example.

$\mathbf{x}_i \in \mathbb{R}^p$ is missing, it will be ignored and all computations will be accomplished using the remaining available values from $\mathbf{x}_i$. A convenient way to indicate the available data is to define a projector (c.f., the indicator matrix by Little and Rubin [258]), which separates the missing and available values in the following way:

$$(\mathbf{p}_i)_j = \begin{cases} 1, & \text{if } (\mathbf{x}_i)_j \text{ exists,} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

By further denoting $\mathbf{P}_i = \text{Diag}\{\mathbf{p}_i\}$, for example, the l_q-norm can be redefined as

$$l_q^{\varnothing}(\mathbf{x}_i) = \|\mathbf{P}_i\mathbf{x}_i\|_q = \left(\sum_{j=1}^{p} |(\mathbf{p}_i)_j(\mathbf{x}_i)_j|^q \right)^{1/q}, \quad 1 \leq q \leq \infty. \tag{10}$$

The Gower's general similarity coefficient [147] or the one given by (6) can be used to implement the available value strategy into clustering algorithms [106, p.53]. Let us consider two p-dimensional objects $\mathbf{x}_i$ and $\mathbf{x}_j$. The weight of the components that are missing in either or both of the objects is zero, and the similarity is computed as the average of the remaining variables. Kaufman et al. [220] define a distance measure with a treatment of incomplete cases for the clustering algorithm CLARA.

Imputation Imputation methods fill in the missing data with values that are estimated using the available data and known relationships [27, 106]. Finding the best estimate for a missing value is not a straightforward task. It depends, for instance, on those statistical measures that must remain unbiased. Additionally, imputation may have an unequal effect on statistical measures of different variables. Hence, it is a complex and case-dependent solution.

The simplest methods are the *case substitution* and *mean* or *mode imputation*. The case substitution method is often used in sample surveys [27]. A data object with missing values is replaced by another non-sampled object. For example, a person that cannot be contacted in a telephone poll can be replaced by another person [27]. The mean and mode imputation are simple and popular methods. The missing data is replaced by mean (quantitative variables) or mode (qualitative variables) of all known values of that attribute. The mean imputation may work when the sample is drawn from a unimodal normal distribution and data is missing at random. However, the problem in this case is that the imputed data will lead to underestimated variance even if the model used to generate the

imputations is correct, because the mean value does not contribute to the variance [258]. Therefore, the mean imputation affects the correlation between the imputed and any other variable. From the cluster analysis point of view, the problem with these simple imputation methods is that the potential class structure of the data is not considered in the operation. Shrinking the within-clusters variances may lead to problems in the cluster validation. Information about the variability of the missing data estimators and uncertainty about the correct model could be assigned to the imputed values by using *multiple imputation* methods. In order to achieve unbiased statistics for clusters, the imputation should be done separately for each homogeneous group in the data. A summary over whole data may be misleading. Hence, cluster analysis, especially from the data mining point of view, needs more sophisticated imputation methods, because the homogeneous groups, that is clusters, are not known in advance.

Hot deck and *cold deck methods*, and *prediction models*, are more appropriate methods for clustered data than the previous imputation methods. The hot deck imputation method substitutes missing data with values that are based on the estimated missing data distributions. The missing data distributions are estimated from the available data values. The cold deck approach differs from the hot deck in that the fill-in values are taken from some other data source. A typical hot deck procedure proceeds in two steps [27]. In the first step, data objects are partitioned into clusters using the available data. In the second step, if incomplete data objects have not been assigned yet, they are assigned to the best-fitting clusters or, otherwise, the missing values of the assigned clusters are filled in with the cluster-wise estimates (such as mean or mode of a cluster). Algorithm 3.2.1 presents an iterative hot deck imputation procedure by Everitt et al. [106].

Algorithm 3.2.1. Iterative hot deck imputation

Step 1. (*Complete-case clustering*) Cluster the complete cases of the incomplete data set.

Step 2. (*Assignment*) Assign the incomplete data objects to the nearest clusters.

Step 3. (*Imputation*) Impute the missing data values using statistical summaries based on within-cluster data.

Step 4. (*Clustering*) Perform clustering for the data set using the observed and imputed data values. If there are changes in clusters, repeat from Step 3.

Algorithm 3.2.1 is actually closely connected to the well-known *EM algorithm*, which is an iterative algorithm for computing the maximum likelihood estimates for model parameters from incomplete data [85, 258]. Each iteration of the EM-algorithm consists of two steps: E-step (expectation step) and M-step (maximization step). The E-step estimates and replaces the missing values assuming that the current parameter estimates are correct. The M-step re-estimates the parameters for the model as if there were no missing data. These steps are iterated until convergence is achieved. The EM-algorithm is a very general approach that can be applied to a broad range of problems including missing data

situations, such as mixture models (clustering models [41]), variance component estimation, iteratively re-weighted least squares, and so on. The K-means clustering algorithm can also be considered as a simplified variant of the EM-principle.

Similar Response Pattern Imputation (SRPI) [292] and *nearest neighbor hot deck* [258, p.65] are so-called matching methods that identify candidates, which are the most similar objects to the object with missing data and substitute the missing data of the incomplete object with values of the candidate object. The candidate has to be complete in the required variables and the distance between the incomplete and the candidate objects must be less than some limit value d_0. As variables are standardized, the l_2- or l_∞-norms, for example, can be used for the similarity comparison. Because some of the values are missing from the compared objects, distance must be first computed as given in (9). Also the general distance measure given in Section 3.2.6 can be used. If more than one of the data objects minimize the chosen distance measure to the incomplete object, then the average value can be used for substitution of the missing variable. On the other hand, if the distance between the candidate and the incomplete object is more than the limit d_0, the imputation will be omitted. This prevents to some extent the occurrence of strange data objects during imputation. Nevertheless, this approach could be useful from cluster analysis point of view due to the local nature of the imputation procedure. The weakness is the required computational power on large data sets that are unavoidable in data mining context.

Prediction model methods use predictive models to estimate values that will be used as substitutes for missing data [27]. The variable with missing data is used as a class-attribute and the remaining data is used as an input for the predictive model. Depending on the type of the variable with missing data (nominal or continuous), either classification or regression model can be used. This approach requires that there are correlations among the variables or, otherwise, the obtained values are not precise for estimating the missing data. This approach leads easily to more well-behaving values for missing data than the true values would be.

3.2.9 Clustering algorithm

> "*We argue that there are many clustering algorithms, because the notion of "cluster" cannot be precisely defined.*"

The above argument is taken from the Estivill-Castro's article: *Why so many clustering algorithms* [99]. Hundreds of clustering algorithms have been developed for analyzing inherent structures of various data sets. Each clustering algorithm is based on a set of underlying assumptions and criteria, and thereby the methods are distinctly biased. Hence, it is obvious that different methods produce different solutions. For example, a hierarchical single linkage algorithm is prone to find elongated clusters, whereas the K-means methods favor hyperspherical clusters. Aldenderfer et al. [8] state that sometimes the properties of a chosen clustering method have even more effect to the result than the data itself.

An extensive survey on clustering algorithms is given by Xu et al. [400]. Other comprehensive sources are, e.g., [106, 203, 149, 30]. A number of classifications have been created for clustering algorithms. A class name describes often algorithmic features (hierarchical, grid-based, neural networks, ...) or properties of the intended applications (e.g., large-scale, mixed data, ...). At least the following classes of clustering algorithms exist: partitioning (e.g., K-means [265]), hierarchical (agglomerative and divisive algorithms, e.g., [204]), density-based [98, 342, 187, 12, 238], grid-based [65, 3, 348, 392], constraint-based [376], fuzzy [21, 22, 176], mixture-densities [124, 126], graph theory-based [312], neural networks-based [19, 20], aggregation-based [143], kernel-based [224], evolutionary methods [285, 18], multi-objective [244], large-scale clustering [41, 418, 150, 388], etc. A positive thing in having a lot of various clustering algorithms is that different methods reflect different aspects of the data and, thereby, give a many-sided view into their internal structure. This is utilized especially in the clustering aggregation method by determining the optimal clustering solution as the one that minimizes the disagreements between several clustering solutions [143]. Han et al. [167] divide the algorithms into five classes: partitioning, hierarchical, density-based, grid-based, and model-based methods. Bradley et al. [44] propose a categorization of three classes: metric-distance based methods, model-based methods, and partition-based methods. Also Hastie et al. [175] divide clustering algorithms into three distinct classes: combinatorial algorithms, mixture modelling, and mode seeking. Estivill-Castro states in [99] that the strongest distinction among different clustering algorithms is between mathematical (continuous) and structural (discrete) models.

Despite of several algorithm classes, two most commonly used approaches are hierarchical and partitioning algorithms. These are also the most traditional methods. The focus of this work is on partitioning methods. The algorithm classes are not strictly separated and some methods include features of two or more classes. For instance, density-based method DENCLUE [187] utilizes grids for initialization. A hierarchical method and Gaussian-based mixture models are combined by Fraley et al. [124]. Evolutionary methods may include a partitioning [285, 18] and hierarchical [138] approach. Fuzzy partitioning and graph theory clustering is combined in [325]. A kernel-based fuzzy partitioning clustering method for incomplete data is proposed in [416].

Since different algorithms produce different type of clusterings, there is no sense to make straight comparison on the quality of the obtained clusterings. Whereas the K-means algorithm finds only spherical clusters, density-based algorithms are capable of finding clusters with arbitrary shapes. Hence, these methods have different goals and rather complement than compete against each other.

3.2.10 Number of clusters

The problem of determining the correct number of clusters in a data set is perhaps the most difficult and ambiguous part of cluster analysis. One of the most fundamental reasons for this is the non-unique nature of the overall clustering

problem, which means that there are often more than one valid solution for a given clustering problem. It was shown in Figure 5 that the ”true” number of clusters depends on the ”level” one is viewing the data. Another problem is due to the methods, that may yield the ”correct” number of clusters for a ”bad” classification [173]. Furthermore, it has been emphasized that mechanical methods for determining the optimal number of clusters should not ignore the fact that the overall clustering process has an unsupervised nature and its fundamental objective is to uncover the unknown structure of a data set, not to impose one (c.f., Anderberg in [9, p.15] and Everitt et al. in [106, p.7-8]). For these reasons, one should be well aware about the explicit and implicit assumptions underlying the actual clustering procedure before the number of clusters can be reliably estimated or, otherwise, the initial objective of the process may be lost. As a solution for this, Hardy [173] recommends that the determination of the optimal number of clusters should be made by using several different clustering methods that together produce more information about the data (cf. clustering aggregation [143] and the mixture of experts model in neural computation [181]). By forcing a structure to a data set, the important and surprising facts about the data will likely remain uncovered.

In some applications the number of clusters is not a problem, because it is predetermined by the context [175]. Then the goal is to obtain a mechanical partition for a particular data using a fixed number of clusters. Such a process is not intended for inspecting new and unexpected facts arising from the data. Hence, splitting up a homogeneous data set in a ”fair” way is much more straightforward problem when compared to the analysis of hidden structures from heterogenous data sets. An illustrative example is the partitioning of a country into a number of telephone areas [220]. Or a company that is going to share out the customer database for K sales persons such that the customers in the group of each sales person are mutually as similar as possible. Obviously the number of clusters is determined by the number of sales persons and partitioning into K clusters will be performed, no matter whether there is K homogenous groups in the database or not. Hence, the principal goal of these clustering problems is not to uncover novel or interesting facts about data. These examples show that the need for methods that also determine the number of clusters depends on the application.

Numerical methods can usually provide only guidance about the true number of clusters and the final decision is often an *ad hoc* decision that is based on prior assumptions and domain knowledge. Therefore, the choice between the different numbers of clusters is often made by comparing several alternatives, and the final decision is a subjective problem that can be solved in practice only by humans. Nevertheless, a number of methods for objective assessment of cluster validity have been developed and proposed. Because the recognition of cluster structures is difficult, especially in high-dimensional spaces, various visualization techniques can also be of valuable help to the cluster analysts.

Many different methods for determining the number of clusters have been developed. Hierarchical clustering methods provide direct information about the

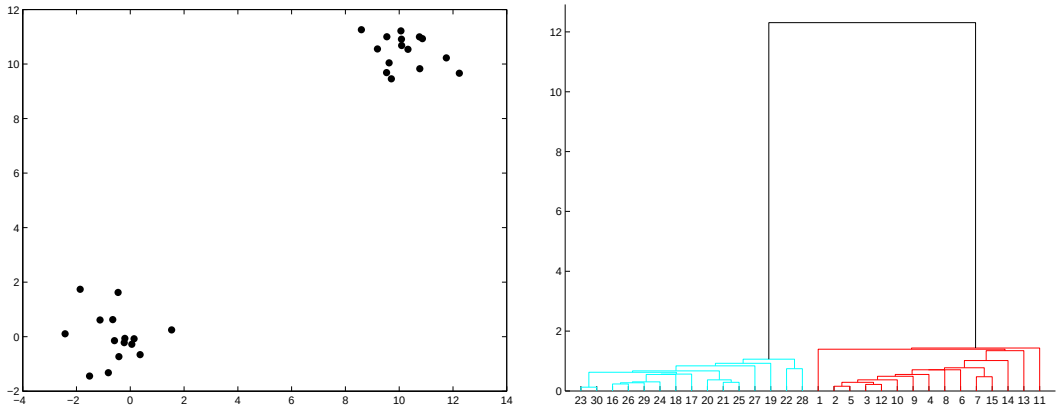


FIGURE 10 Left: a bimodal data set ($n = 30$). Right: Dendrogram tree (see, e.g., [106, p.55]) after the hierarchical single-linkage clustering.

number of clusters by clustering the objects on a number of different hierarchical levels, which are then presented by a graphical tree structure known as *dendrogram* (see Figure 10). One may apply some external criteria to validate the solutions on different levels or use the dendrogram visualization for determining the best cluster structure.

Partitioning-based clustering methods, for example, K-means [265] and PAM [220], take the number of clusters as an input parameter. Hence, the algorithm should be run several times for different number of clusters and the best number should be chosen by some external criteria. Some modified variants of the partitioning methods exist that are based on splitting and merging rules in order to increase or decrease the number of clusters as the algorithm proceeds. Examples of such methods are ISODATA and the coarsening-refining approach of K-means.

The selection of the correct number of clusters is actually a kind of model selection/validation problem. A large number of clusters provides a more complex "model" whereas a small number may approximate data too much. Hence, several methods and indices have been developed for the problem of cluster validation and selection of the number of clusters, see, e.g. [340, 95, 91, 283, 9, 106, 129, 173, 236, 207, 371, 329, 161, 314, 361, 366]. Many of them are based on the within- and between-group distance. Generally, there are three main approaches to the cluster validation [156]. *Internal criteria* utilize learning data in the validation. *External criteria* require test data on which to validate the goodness of the obtained clustering solution. *Relative criteria* compares the obtained clustering to another clustering structure that is obtained by the same algorithm, but with different initial parameters. A more detailed treatment for cluster validity and the problem of the unknown number of clusters is given later in Chapter 8. Moreover, some examples on real-world data will also be presented.

3.2.11 Interpretation of results

Interpretation of the clustering results is often performed using visualization techniques. At this point, available domain knowledge can be integrated in the obtained clustering solution. In two- or three dimensions the visualization is straightforward, but in higher dimensions projection and transformation techniques may be necessary. The most common approaches are the PCA and MDS techniques. These are treated in more detail in Chapter 8.

3.3 Partitioning-based clustering algorithms

Perhaps the most popular class of clustering algorithms are combinatorial optimization algorithms a.k.a iterative relocation algorithms. They minimize a given clustering criterion by iteratively relocating data points between clusters until a (locally) optimal partition is attained. In basic iterative algorithms, such as K-means- or K-medoids, convergence is always local. Because the number of data points in any data set is always finite and, thereby, also the number of distinct partitions is finite, the problem of local minima could be avoided by searching the globally best solution with an exhaustive search method. This is achievable in theory, but finding the globally optimal partition is known to be an NP-hard problem and the exhaustive approach through all partitions is not useful in practice [93, p.226]. The number of different partitions for n observations into K groups is a Stirling number of the second kind, which is given by

$$S_n^{(K)} = \frac{1}{K!}\sum_{i=0}^{i=K}(-1)^{K-i}\binom{K}{i}i^n.$$

This shows that the enumeration of all possible partitions is practically impossible even for relatively small problems. The problem is further exacerbated when the number of clusters is unknown. In that case the number of different combinations is the sum of the Stirling numbers of the second kind:

$$\sum_{i=1}^{i=K_{max}} S_n^{(i)},$$

where K_{max} is the maximum number of clusters for which it obviously holds that $K_{max} \leq n$. The fact is that exhaustive search methods are far too time-consuming even for modern computing systems. Moreover, it seems to be an infinite race between the computer power and the amount of data, which both have increased rapidly during the last years. Therefore, iterative optimization is a more practical approach than exhaustive search.

3.3.1 Iterative relocation algorithm

Iterative optimization clustering starts with an initial partition. The quality of this partition is then improved by applying a local search algorithm to the data. Sev-

eral methods of this type are often categorized as a partitioning cluster method (a.k.a. non-hierarchical or flat methods [94]). A general iterative relocation algorithm, which provides a baseline for partitioning-based clustering methods, is given in Algorithm 3.3.1 (see, [168],[106, pp.99-100] or [8, p.45]).

Algorithm 3.3.1. Iterative relocation algorithm

Input: The number of clusters K, $n \times p$ data set $\mathbf{X}$.

Output: A set of K clusters, which minimizes a criterion function $\mathcal{J}$.

Step 1. (*Initialization*) Begin with the initial K cluster centers/distributions as the initial solution.

Step 2. (*Recomputation*) (Re)compute memberships for the data points with respect to the current cluster centers.

Step 3. (*Update*) Update some/all cluster centers/distributions according to the new memberships of the data points.

Step 4. (*Stopping rule*) Repeat from Step 2. until there is no change to $\mathcal{J}$ or no data points change cluster.

Using this framework, iterative methods compute the estimates for cluster centers, which are often referred to as prototypes or centroids. The prototypes are meant to be the most representative points for the clusters. The mean and median are typical choices for the estimates. On the other hand, some methods, such as the EM-algorithm [41, 85], estimate a set of parameters that maximizes the likelihood of the chosen distribution model for a data. The best-known of the prototype-based algorithms are K-means and K-medoids, whereas the EM-algorithm is probably the most popular distribution-based algorithm [168]. The methods differ in the way they represent clusters, but they are mostly based on the same general algorithmic principle, which is given by Algorithm 3.3.1. The medoid-based algorithms are needed in special applications that include restrictive domain structures (such as Web navigation paths, see for example, [103] and Chapter 2) whose integrity must be retained. K-means is discussed more thoroughly later in this work.

In summary, there are three basic elements that can be easily modified in the general relocation algorithm: 1. Initialization, 2. Reassignment of data points into clusters, and 3. Update of the cluster parameters. Although the heuristical, computational and statistical properties of iterative partitioning methods are mainly defined through the realization of these elements, there are also other influencing factors, such as treatment of missing data values, that effect the overall behavior.

Initialization

Due to the non-convex nature of criterion functions to be minimized, the iterative relocation methods are often trapped into one of the local minima. This makes

the quality of final clustering solutions dependent on the initial partition. A simple approach is to run the partitioning-based algorithms by starting from several initial conditions. Another, more sophisticated way is to use some heuristic for finding an optimal initialization. In general, the initialization of the partitioning-based clustering algorithms is an important matter of interest (previous studies, e.g., [9, 273, 315, 43, 218, 222, 6, 182, 119, 365]).

Main iteration

The reassigment of data points and the update of prototypes (or parameters) construct the pass through data that improves the quality of the clustering. There are two main types of passes: *nearest centroid sorting pass* (a.k.a. K-means pass) and *hill-climbing pass* [8] or [9, p.160-162]. Let us refer to the passes by NCS-pass and HC-pass, respectively.

The NCS-pass simply assigns data points to the cluster with the nearest prototype. Aldenfelder [8] divides the NCS-passes into *combinatorial* and *noncombinatorial* cases. In the former case, cluster centers are recomputed immediately after the reassigment of a data point (c.f. MacQueen's K-means and its variant [9]). In the latter case, the cluster centers are recomputed only after all data points are reassigned to the closest cluster centers (c.f. Forgy's K-means and Jancey's variant [121, 9]). The NCS-pass implicitly optimizes a particular statistical criterion (e.g., $\mathrm{tr}(\mathbf{W})$ for K-means) by moving data points between the clusters, whereas the HC-pass moves the points from a cluster to another only if the move improves the value of the criterion function.

Problem of unknown number of clusters

The name of a partitioning-based clustering method is usually of the form *K*-"estimates" (sometimes, mostly in the context of fuzzy clustering, also *C*-"estimates" is used, see [351, 21, 22, 400], and articles therein), which is due to the tendency to partition a data set into a fixed number (K) of clusters. Another well-known class of clustering algorithms, namely hierarchical algorithms, produce a set of solutions with different numbers of clusters, which are then presented visually by a hierarchical graphical structure called dendrogram (see Figure 10). Although hierarchical methods provide some information about the number of clusters, they are not very feasible for data mining problems. First, quadratic memory requirement of the dissimilarity matrix is intractable for large data sets. Secondly, construction of the dissimilarity matrix is troublesome for incomplete data, because distances between data points lying in different subspaces are not directly comparable. This opens up another interesting problem: estimation of the correct number of clusters for partitioning-based algorithms. Typically, K is estimated using some measure to compare the validity of the clustering solution (see Section 3.2.10).

3.3.2 K-means clustering

Basically *K-means* is an iterative process that divides a given data set into K disjoint groups. K-means is perhaps the most widely used clustering principle, and, in particular, the best-known of the partitioning-based clustering methods that utilize prototypes for cluster presentation (a.k.a representative-based algorithm by Estivill-Castro [99]). It is also employed, by the name *Lloyd's algorithm*, as a vector quantization technique for signal compression problems [140]. Quality of K-means clustering is measured through the within-cluster squared error criterion (e.g., [9, p.165] or [175])

$$\begin{aligned} &\min_{\mathbf{c}\in\mathbb{N}^n,\mathbf{m}_k\in\mathbb{R}^p} \mathcal{J}(\mathbf{c},\{\mathbf{m}_k\}_{k=1}^K) = \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{m}_{(\mathbf{c})_i}\|_2^2 \\ &\text{subject to } (\mathbf{c})_i \in \{1,\ldots,K\} \text{ for all } i = 1,\ldots,n, \end{aligned} \tag{11}$$

where $\mathbf{c}$ is a code vector, which represents the cluster assignments of the objects, and $\mathbf{m}_{(\mathbf{c})_i}$ is the mean of the cluster, where the data point $\mathbf{x}_i$ is assigned to. The sample mean leads to a unique minimum of the within-cluster variance, from which it follows that the problem actually corresponds to the minimization of $\sum_{i=1}^{K} \mathrm{tr}(\mathbf{W}_i)$, where $\mathbf{W}_i$ is the within-group covariance matrix of the i^{th} cluster. Thus, the K-means clustering is also referred to as a variance minimization technique [220, p.112]. Actually in 1963, before the invention of the first K-means algorithms, the minimum variance optimization technique was used by Ward [393], who proposed a hierarchical algorithm that begins with each data points as its own cluster and proceeds by combining points that result in the minimum increase in the sum of squares error value.

As such, K-means clustering tends to produce compact clusters, but it does not take into account the between-cluster distances. The use of the squared l_2-norm makes the problem formulation extremely sensitive towards large errors, which means that the formulation is non-robust in a statistical sense (see, e.g. [195]). However, due to its implementational simplicity and computational efficiency, K-means has retained its position as an extremely popular principle for many kind of cluster analysis tasks. It also requires less memory resources than, for instance, hierarchical methods. By courtesy of its computational efficiency, K-means is also applied to initialization of other more expensive methods (e.g., EM-algorithm [38, 43]). The K-means algorithm, which is used to minimize the problem of K-means (12), has a large number of variants which are described next.

K-means algorithms

K-means type grouping has a long history. Already in 1958 Fisher [120] investigated this problem in a one-dimensional case as a *grouping problem*. At that time, algorithms and computer power were still insufficient for larger-scale problems, but the problem was shown to be interesting with concrete applications. Hence,

procedures more efficient to exhaustive search were needed. The seminal versions of the K-means procedure were introduced in the Sixties by Forgy [121] (cf. discussion in [265]) and MacQueen [265] (see also [9] and [30]). These are perhaps the most widely used versions of the K-means algorithms [220, p.112]. In 1982, Lloyd [260] presented a quantization algorithm for the problem of pulse-code modulation (PCM) for analog signals. The algorithm is often referred to as the Lloyd's algorithm and, actually, it is equivalent with the Forgy's K-means algorithm in a scalar case. Although Lloyd's paper was not published until 1982, the unpublished manuscript from 1957 is referred, for example, in articles from 1977 and 1980, by Chen [68] and Linde et al. [255], respectively[2]. A basically similar algorithm for multidimensional cases was presented by Chen in [68]. Linde et al. generalized the Lloyd's algorithm to a vector quantization algorithm [255]. This is often referred to as the *Generalized Lloyd's Algorithm* (GLA) in signal and image processing context. Hence, there are two main types of K-means algorithms that have actually been discovered more than once. The main difference between the Forgy's and MacQueen's algorithms is the order in which the data points are assigned to the clusters and the cluster centers are updated. The MacQueen's K-means algorithm updates the "winning" cluster center immediately (online) after every assignment of a data point and all cluster centers one more time after all data points have become assigned to the clusters, while the Forgy's method updates the cluster centers only after all data points are assigned to the closest cluster centers. Moreover, the algorithm iterates until converged while the MacQueen's algorithm does not iterate down to convergence. It performs only one complete pass through data. The starting points are often the first K data points in the data set.

Next the MacQueen's K-means algorithm, its convergent variant, and the Forgy's algorithm are given. The following notations are used:

- p: number of dimensions
- n: number of data points
- $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$: given p-dimensional data set
- K: number of clusters
- $\mathcal{C}_1 \cup \mathcal{C}_2 \cup \ldots \cup \mathcal{C}_K$: cluster sets
- n_k: size of k^{th} cluster
- $\mathbf{m}_1, \ldots, \mathbf{m}_K \in \mathbb{R}^p$: cluster centers (prototypes)
- $\mathbf{c} \in \mathbb{N}^n$, $\{1 \leq (\mathbf{c})_i \leq K\}$: $n \times 1$ code vector (cluster assignments)
- maxit: maximum number of iterations
- t: iteration count

[2] According to the author's note [260], the manuscript of this article was written and circulated for comments at Bell Laboratories already in 1957.

MacQueen's K-means algorithm First, we present the MacQueen's single pass variant of the K-means algorithm. This variant is relatively fast to compute, because the algorithm performs only one pass through the complete data set and finalizes the cluster means one more time in the end. This also ensures that the predefined number of clusters can not change during the process when each cluster is initialized with at least one data point. According to Anderberg [9, p.163], acceptance of the first reallocation of data points should give apparently good results, because the consequent reallocations usually result in relatively few assignments. However, the quality of the obtained partitions is questionable and the results also depend on the sequence in which the data points are processed. Moreover, due to its single pass nature one can not speak about convergence.

Algorithm 3.3.2. MacQueen's (online) K-Means algorithm

Required input parameters: $\mathbf{X}$ and K.

Optional input parameters: $\{\mathbf{m}_k\}_{k=1}^{K}$.

Output parameters: $\{\mathbf{m}_k\}_{k=1}^{K}$ and $\mathbf{c}$.

Step 1. (*Initialization*) If needed, choose the initial cluster centers (for example, set $\mathbf{m}_i = \mathbf{x}_i$ for $i = 1, \ldots, K$). Set $n_k = 0$ for $k = 1, \ldots, K$.

Step 2. (*Main iteration*) For each point $\mathbf{x}_i$ ($i = 1, \ldots, n$)

1. Step 2.a. (*Assignment*) Assign $\mathbf{x}_i$ to its closest cluster C_k by

$$(\mathbf{c})_i = \arg\min_k \|\mathbf{x}_i - \mathbf{m}_k\|_2.$$

   Set $n_k = n_k + 1$.
2. Step 2.b. (*Update*) Recompute the center of the gaining cluster $\mathcal{C}_k$ as

$$\mathbf{m}_{(k)} = \frac{1}{n_k} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \mathbf{x}_i.$$

Step 3. (*Recomputation*) Update the centers of the final partition by recomputing for all $k \in \{1, \ldots, K\}$

$$\mathbf{m}_{(k)} = \frac{1}{n_k} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \mathbf{x}_i.$$

Convergent variant of MacQueen's K-means algorithm In [9], Andeberg proposes the following variant for the MacQueen's algorithm that iterates until convergence. Due to the reassignment rule this variant is also dependent on the order of the data points.

Algorithm 3.3.3. Convergent MacQueen's K-Means algorithm

Required input parameters: $\mathbf{X}$, K, and maxit.

Optional input parameters: $\mathbf{c}^0$.

Output parameters: $\{\mathbf{m}_k\}_{k=1}^K$ and $\mathbf{c}$.

Step 1. (*Initialization*) If needed, partition data into initial clusters (for example using the main iteration of the ordinary MacQueen's algorithm). Set $t = 0$.

Step 2. (*Main iteration*) For each point $\mathbf{x}_i \in \mathbf{X}$ $(i = 1, \ldots, n)$

1. Step 2.a. (*Assignment*) Assign $\mathbf{x}_i$ to a new cluster $\mathcal{C}_r^t$ if

$$\min_{r \neq (\mathbf{c}^t)_i} \|\mathbf{x}_i - \mathbf{m}_r^t\|_2 < \|\mathbf{x}_i - \mathbf{m}_{(\mathbf{c}^t)_i}^t\|_2.$$

2. Step 2.b. (*Update*) Recompute the centers of the losing and gaining clusters, $\mathcal{C}_{(\mathbf{c}^t)_i}^t$ and $\mathcal{C}_r^t$, respectively, as

$$\mathbf{m}_{(\mathbf{c}^t)_i}^t = \frac{1}{n_{(\mathbf{c}^t)_i} - 1}(n_{(\mathbf{c}^t)_i}\mathbf{m}_{(\mathbf{c}^t)_i}^t - \mathbf{x}_i)$$

and

$$\mathbf{m}_r^t = \frac{1}{n_r^t + 1}(n_r^t \mathbf{m}_r^t + \mathbf{x}_i).$$

Set $n_r = n_r + 1$, $n_{(\mathbf{c}^t)_i} = n_{(\mathbf{c}^t)_i} - 1$, $(\mathbf{c}^t)_i = r$, and $t = t + 1$.

Repeat Step 2. until convergence is achieved; that is, continue until the full cycle through the data set fails to cause any changes in cluster memberships or $t =$ maxit.

Forgy's K-means algorithm The next algorithm, Forgy's batch type algorithm, is at present perhaps the most widely used version. It is independent of the order of the data points. However, it is also a local procedure, which means that the solution depends on the initial conditions. A trouble for a clustering task may arise from the possibility of ending up with less than K clusters.

Algorithm 3.3.4. Forgy's (batch) K-Means algorithm

Required input parameters: $\mathbf{X}$, K, and maxit.

Optional input parameters: $\{\mathbf{m}_k^0\}_{k=1}^K$ or $\mathbf{c}^0$.

Output parameters: $\{\mathbf{m}_k^*\}_{k=1}^K$ and $\mathbf{c}^*$.

Step 1. (*Initialization*) If centers $\{\mathbf{m}_k^0\}_{k=1}^K$ are given then go to Step 2. Else if an initial partition $\mathbf{c}^0$ is given then go to Step 3. If neither centers nor partition is given as input then initialize centers $\{\mathbf{m}_k^0\}_{k=1}^K$, assign each data point $\{\mathbf{x}_i\}_{i=1}^n$ to the closest center and go to Step 3. Set $t = 0$.

Step 2. (*Reassignment*) Reassign each data point in $\{\mathbf{x}_i\}_{i=1}^n$ to a new cluster $\mathcal{C}_r^t$, if

$$\min_{r \neq (\mathbf{c}^t)_i} \|\mathbf{x}_i - \mathbf{m}_r^t\|_2 < \|\mathbf{x}_i - \mathbf{m}_{(\mathbf{c}^t)_i}^t\|_2.$$

Set $(\mathbf{c}^t)_i = r$ and update n_k^t for all $k = 1, \ldots, K$.

Step 3. (*Recomputation*) Update the cluster centers of the current partition by recomputing for all $k = 1, \ldots, K$

$$\mathbf{m}_{(k)}^t = \frac{1}{n_k^t} \sum_{\mathbf{x}_i \in \mathcal{C}_k^t} \mathbf{x}_i.$$

Step 4. (*Stopping*) If no reassignments of data points between cluster centers occur or $t =$ maxit, then stop. Otherwise, set $t = t + 1$ and repeat from Step 2.

A convergence proof for the algorithm is given by Selim et al. [344]. The time complexity of the Forgy's K-means is $\mathcal{O}(npKt)$ (t is the number of iterations) [94]. This is reasonable from the DM point of view, since n is usually significantly greater than p, K, or t [102]. Because of the algorithmic details, the MacQueen's and Forgy's algorithms are also referred to as online- and batch-K-means algorithms, respectively (see, e.g., [38, 347]).

For example, in [38, 347], the convergent variant (Algorithm 3.3.3) of MacQueen's K-means algorithm is behind the online clustering, although the MacQueen's K-means algorithm is referred. In [38], the numerical experiments suggest that the online K-means algorithm converges faster during the first few passes through the data and, thereafter, batch version (Algorithm 3.3.4) outperforms it. However, the online clustering may be useful in real-time applications, which have to respond to inputs in extremely short time, or receive the data in a stream of unknown length, or if there is not enough memory available to store a data set as a complete block [21].

Drawbacks

Despite the wide popularity of the ordinary K-means algorithms, there are some significant defects that have led to the development of numerous alternative versions during the past years (see, e.g., [315, 43]):

- *Sensitivity to initial configuration.* Since the basic algorithms are local search heuristics and K-means cost function is non-convex, it is very sensitive to the initial configuration, and the obtained partition is often only suboptimal (not the globally best partition).

- *Lack of robustness.* The sample mean is very sensitive to outliers. So-called breakdown point is zero, which means that a single gross error may distort the estimate completely. The obvious consequence is that the K-means problem formulation is non-robust as well.

- *Unknown number of clusters.* Since the algorithm is a kind of "flat" or "non-hierarchical" method [94], it does not provide any information about the number of clusters.

- *Empty clusters.* The Forgy's batch version may lead to empty clusters on unsuccessful initialization.

- *Order-dependency.* MacQueen's basic and converging variants are sensitive to the order in which the points are relocated. This is not the case for the batch versions.

- *Only spherical clusters.* K-means presumes the symmetric Gaussian shape for cluster density functions. From this it follows that a large amount of clean data is usually needed.

- *Handling of nominal values.* The sample mean is not defined for nominal values.

- *Method is statistically biased and inconsistent.* The method converges often to the wrong parameter values (local optimum of poor quality) even if provided with the correct number of clusters (distributions) and the data exist in large amounts following multivariate Gaussian distributions [102].

In order to solve the previous problems many variants for the original versions have been developed.

Enhanced variants of K-means algorithm

It seems that the development of the clustering algorithms was very intensive during the sixties. As we know, the rapid development of PC computer systems during the eighties and still growing data storages led to the invention of knowledge discovery and data mining concepts. It seems that this development has led again to the growing interest in clustering algorithms. Hence, many variants for the traditional K-means algorithms have emerged during the last ten years. Many of these try to solve the known drawbacks of the K-means procedures.

The general version of the iterative relocation algorithm (Algorithm 3.3.1) provides a great number of optional elements to be implemented in different ways when building an iterative relocation algorithm for solving the problem of K-means. First, there are many ways to generate an initial partition or cluster prototypes for a data set. Second, there are many ways to arrange the relocation of the data points and update of the prototypes (for example, see [106, p.100]). The data points can be assigned to the nearest cluster or to the one that leads to the largest reduction in the value of the objective function. The cluster prototypes can be updated, either after every single reassignment of a data point, or after a fixed number of reassignments. Therefore, it is not surprising that the K-means clustering algorithm receives a somewhat many-sided treatment in the clustering literature. Although the differences among the variants do not seem to be

remarkable, the algorithms may sometimes produce different final partitions for the same data despite starting with the equal initial conditions.

Together with the basic K-means algorithm, MacQueen [265] presented a "coarsening–refining" variant that estimates also the correct number of clusters. It starts with a user specified number of clusters, and then coarsens and refines clustering according to input parameters during the process. After each assignment, all pairs of cluster means whose distance to each other is less than the coarsening parameter will be merged. On the other hand, every time a data point is processed in order to make an assignment to a cluster, its distance to the closest cluster mean is compared with the refining parameter. If the distance exceeds the parameter value, a new cluster will be created.

Another variant that also tries to determine the number of clusters is called ISODATA[3]. This is a quite elaborate algorithm and requires a lot of inputs from the users. If the user is looking at the data from data mining perspective, which means a minimal amount of prior assumptions and information, the use of this algorithm may prove to be complicated.

Jancey's variant is a modification for the Forgy's K-means method [9, p.161–162], which is expected to accelerate convergence and avoid inferior local minima. In this variant the new cluster center is not the mean of the old and added points, but the new center is updated by reflecting the old center through the mean of the new cluster.

In order to avoid poor suboptimal solutions, a number of different initialization methods for K-means(-type) methods have been developed and evaluated through numerical experiments (cf., the references in Section 3.3.1). Zhang et al. [413] suggested to run the so-called *K-Harmonic Means* algorithm prior to K-means. They reported that in comparison to the K-means algorithm, K-Harmonic Means is more insensitive to initial configurations, but it converges slower near the solution. An accelerated, *kd*-tree-based variant for the K-means clustering is proposed by Pelleg et al. [313]. The authors suggest this method for initialization of the ordinary K-means algorithm as well. As a problem the authors report the scalability with respect to the number of dimensions, which is due to the use of the *kd*-tree structure. One of the most interesting approaches to avoid poor quality minima in clustering problems is the *LBG-U method*, which is presented as a vector quantization method [134]. Since the LBG-algorithm [255] is equivalent with the Forgy's K-means clustering algorithm, the LBG-U method can also be used as a clustering method. The idea of the LBG-U is to repeat the LBG-algorithm until convergence. After each run, the cluster center with the minimum utility is moved to a new point. The mean vector that possesses the minimum utility is the one that contributes least to the overall sum of squared errors when removed. The new point is chosen close to the mean of the cluster that generates most of the distortion error for clustering. LBG-U is good from the DM point of view, because it does not require more input parameters than the basic K-means algorithms. It also converges, since the algorithm will be terminated if

[3] This is not the same procedure as the the one called Basic Isodata in [93, p.201]. Basic Isodata is actually the same as the Forgy's K-means.

the last run does not produce reduction to the value of the criterion function.

The increased power of the computer systems has enabled the use of more intensive methods in solving the drawbacks of the K-means-type algorithms. Methods based on genetic algorithms based methods have been developed in order to get the globally best solutions for partition problems [239, 18, 263]. In general, genetic algorithms are known as computationally demanding. Kövesi et al. [237] propose a stochastic K-means algorithm that incorporates randomness to the deterministic K-means clustering method. The algorithm is shown to be less dependent on the initial partition than the original K-means, but it requires more computation time. Likas et al. propose *the global k-means* clustering algorithm [254]. The global k-means is a deterministic and incremental global optimization method that employs the K-means procedure as a local search procedure, but is independent of any initial parameters. In order to reduce the computational load of the exhaustive method, a computationally faster variant and a *kd*-tree-based initialization method are also given in the paper.

As the requirements for dealing with data sets, often too large to be loaded into a fast RAM memory even, have been constantly growing, the scalability of the K-means algorithms have become an important issue. A scalable single-pass version of the K-means algorithm, which is based on identification of data that can be compressed, region of data that must be maintained, and regions that can be discarded, is introduced in [45]. An enhanced version for that is proposed in [108]. These methods can be used efficiently for searching multiple solutions from different initial conditions, since the information about the compressible regions is retained, and discarded data can be reused. An efficient "disk-based" algorithm, *Relational K-means*, for clustering large high-dimensional data sets inside a relational database is given in [309]. Disk-based refers to efficient organization of data matrices on the disk.

A number of methods for estimating K, the correct number of clusters, have been developed and experimented with partition-based methods in general. It is not so much an algorithm-specific issue, but a more general problem covering all partition-based methods that are based on solving clustering problems for a specific K. A common approach is to use some validity measure to evaluate the goodness of the obtained solution (cf. the references in Section 3.2.10).

The problem of empty clusters may occur with the Forgy's batch-type algorithms. One cluster center may become empty when all the points are closer to other centers. The risk of empty clusters exists also for other batch-type partitioning clustering methods. This is not the case for the MacQueen's type single pass algorithms. However, on large data sets and with small numbers of clusters, the probability of empty clusters should be small. The problem is worse in sub-sampling based methods due to the smaller number of data points. Therefore, Bradley et al. [43] introduce, as a part of their sub-sampling-based initialization method, a variant of the K-means algorithm, KMeansMod, that never returns empty clusters in the final solution. This method is considered more thoroughly in Chapter 7.

The order-dependency is not a general problem for the batch-variants of

the K-means algorithms, but it is a serious problem for the MacQueen's type online K-means algorithms. The tendency to spherical clusters is an inherent property due to the problem setting. Therefore, prototype-based clustering algorithms tend to construct clusters where the points are close to each other. On the other hand, any kinds of connected areas of arbitrary shape are not of interest. For example, hierarchical (e.g., single-link) and density-based methods are biased to arbitrary non-spherical shape clusters.

Extensions of the K-means-type algorithms to deal with mixed and categorical data types are presented in [326, 191, 192, 154, 341]. The K-modes algorithm by Huang [191] extends the K-means principle to categorical variables by using simple matching coefficients as a dissimilarity measure and replacing the cluster means with modes as cluster representatives. Based on the previous work, Huang [192] proposes a k-prototypes algorithm that further integrates the k-means and k-modes algorithms to allow clustering of objects described by mixed numeric and categorical attributes. Algorithms for clustering binary data streams are represented in [307].

The lack of robustness in K-means algorithms is a consequence of the use of the sensitive square of l_2-norm in the cluster center estimation. The sample mean is straightforward and fast to compute (closed-form solution exists), but it is also extremely sensitive to all gross errors, which inevitably exist in real-world data sets. This makes the usability of the K-means algorithms questionable on noisy and incomplete data sets. This problem is discussed more thoroughly in this thesis.

4 ON NON-SMOOTH OPTIMIZATION AND ROBUST ESTIMATION

In this chapter, the basic elements of non-smooth optimization, nonlinear optimization algorithms, and robust statistics are introduced. The chapter begins by presenting the basic notations that are used throughout the thesis, unless otherwise stated. Basic definitions on convex analysis are given to be used later as grounds for some basic theories of non-smooth optimization. A few nonlinear optimization algorithms and iterative solvers are introduced. These are applied to robust location estimation problems. The basic terminology of the statistical estimation is explained together with an introduction to robust statistics and M-estimation. Finally, two specific multivariate M-estimators, the coordinate-wise and spatial medians, are introduced from the non-smooth optimization perspective with discussions.

4.1 Convex analysis and non-smooth optimization

In this section some basic definitions regarding convex analysis and non-smooth optimization are introduced. The given definitions serve as basic knowledge for the methods and formulations that are used for the computation of the spatial median [267, 305, 33, 28, 332, 74].

4.1.1 Convexity

The concept of convexity is a fundamental and useful property in optimization, and, thereby, in clustering and statistical estimation as well. Theories of convex analysis form the basis for defining optimality conditions for smooth and non-smooth optimization problems. The term *convex* concerns both sets and functions. Hence, let us first define the convex set.

Definition 4.1.1. *A set $S \subset \mathbb{R}^p$ is said to be convex if*

$$\lambda \mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2 \in S$$

whenever $\mathbf{x}_1$ *and* $\mathbf{x}_2$ *are in* S *and* $\lambda \in [0,1]$. *In other words,* S *is convex, if the line joining any two points of the set also belongs to the set.*

Prototype-based partitioning algorithms produce usually clusters with convex geometry, whereas non-convex clusters are obtained, for example, by density-based methods.

The weighted average of the form $\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2$, where $\lambda \in [0,1]$, is known as *convex combination* of $\mathbf{x}_1$ and $\mathbf{x}_2$. Based on this, the definition of the convex hull is given next.

Definition 4.1.2. *Let* S *be a random subset of* $\mathbb{R}^p$. *The convex hull of* S, *denoted by* $conv(S)$, *is the collection of all convex combinations of points on* S.

Convexity of functions is defined by the next definition.

Definition 4.1.3. *Let* $\mathcal{J} : S \rightarrow \mathbb{R}$, *where* S *is a nonempty convex set in* $\mathbb{R}^p$. *The function* $\mathcal{J}$ *is said to be convex on* S, *if*

$$\mathcal{J}(\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \lambda\mathcal{J}(\mathbf{x}_1) + (1-\lambda)\mathcal{J}(\mathbf{x}_2) \tag{12}$$

for each $\mathbf{x}_1, \mathbf{x}_2 \in S$ *and for each* $\lambda \in [0,1]$.

The function $\mathcal{J}$ is called *strictly convex* on S, if strict inequality holds in (12) for all $\mathbf{x}_1, \mathbf{x}_2 \in S$ and for each $\lambda \in [0,1]$. The function $\mathcal{J}$ is called *(strictly) concave* if $-\mathcal{J}$ is (strictly) convex.

4.1.2 Nonlinear optimization

Let us consider nonlinear unconstrained optimization problem of the form

$$\min_{\mathbf{u}\in\mathbb{R}^p} \mathcal{J}(\mathbf{u}). \tag{13}$$

If $\mathcal{J} \in C^1(\mathbb{R}^p)$, (13) is said to be a smooth optimization problem. Such problems can be solved using methods that are based on the classical C^1 calculus. On the other hand, if $\mathcal{J} \notin C^1(\mathbb{R}^p)$, problem (13) is said to be non-smooth [267]. Such problems can not be analyzed (or solved) using classical C^1 calculus. There are two basic approaches to such problems. A non-smooth problem can be smoothed by function approximations so that the classical C^1 assumptions become valid. In the other case, one can use an optimization method that is not based on the C^1 differentiability assumptions. The field of applied mathematics that concentrates on such problems is called non-smooth analysis [267].

Local and global minima

Based on [33, 305] the notion of optimality is described next. Properties of the objective function $\mathcal{J}$ determine the nature of the existing solutions and methods that efficiently find them. The global minimum of $\mathcal{J}$ is a point, where the function attains its minimum value over the whole problem space.

Definition 4.1.4. *A point* $\mathbf{u}^* \in \mathbb{R}^p$ *is a global minimum of* $\mathcal{J}$*, if*

$$\mathcal{J}(\mathbf{u}^*) \leq \mathcal{J}(\mathbf{u}) \text{ for all } \mathbf{u} \in \mathbb{R}^p.$$

If $\mathcal{J}$ is non-convex, its global minimum may be difficult to find. Moreover, the knowledge of $\mathcal{J}$ is usually local. So-called global optimization methods are used for such problems. However, most optimization algorithms attain the locally minimizing points. A local minimum is a point that yields the smallest value of the objective function in its neighborhood.

Definition 4.1.5. *A point* $\mathbf{u}* \in \mathbb{R}^p$ *is a local minimum of* $\mathcal{J}$*, if there exist* δ *such that*

$$\mathcal{J}(\mathbf{u}^*) \leq \mathcal{J}(\mathbf{u}) \text{ for all } \mathbf{u} \in B(\mathbf{u}^*, \delta).$$

The above definition is called *weak*, because the minimum is not necessarily unique in the neighborhood [305]. Hence, the definition of a strict minimum is needed.

Definition 4.1.6. *A point* $\mathbf{u}^* \in \mathbb{R}^p$ *is a strict local minimum of* $\mathcal{J}$*, if there exists* δ *such that*

$$\mathcal{J}(\mathbf{u}^*) < \mathcal{J}(\mathbf{u}) \text{ for all } \mathbf{u} \in B(\mathbf{u}^*, \delta) \text{ with } \mathbf{u} \neq \mathbf{u}^*.$$

If $\mathcal{J}$ is convex, every local minimum is also a global minimum. Local and global maxima are defined correspondingly.

Optimality conditions in smooth problems

As mentioned above, if $\mathcal{J} \in C^1(\mathbb{R}^p)$, optimization problem (13) is said to be smooth. In this case, the minimizing points of $\mathcal{J}$ can be simply characterized by gradient $\nabla\mathcal{J}(\mathbf{u}^*)$. Hence, *the first-order necessary conditions* for optimality are defined by the following theorem [33, 305].

Theorem 4.1.1. *If* $\mathbf{u}^*$ *is a local minimum of* $\mathcal{J}$ *and* $\mathcal{J}$ *is continuously differentiable in an open neighborhood of* $\mathbf{u}^*$*, then* $\nabla\mathcal{J}(\mathbf{u}^*) = \mathbf{0}$*.*

Point $\mathbf{u}^* \in \mathbb{R}^p$ is called a *stationary point*, if $\nabla\mathcal{J}(\mathbf{u}^*) = 0$. Hence, a local minimum is always a stationary point. If $\mathcal{J}$ is moreover twice continuously differentiable in an open neighborhood of $\mathbf{u}^*$, *the second-order necessary optimality conditions* are defined by the following theorem [33, 305].

Theorem 4.1.2. *If* $\mathbf{u}^*$ *is a local minimum of* $\mathcal{J}$ *and* $\nabla^2\mathcal{J}$ *is continuous in an open neighborhood of* $\mathbf{u}^*$*, then* $\nabla\mathcal{J}(\mathbf{u}^*) = \mathbf{0}$ *and* $\nabla^2\mathcal{J}(\mathbf{u}^*)$ *is positive semidefinite.*

The necessary optimality conditions ascertain only the local optimality. Strict local optimality is guaranteed by *the second-order sufficient optimality conditions* that are defined by the following theorem [33, 305].

Theorem 4.1.3. *Suppose that* $\nabla^2\mathcal{J}$ *is continuous in an open neighborhood of* $\mathbf{u}^*$ *and that* $\nabla\mathcal{J}(\mathbf{u}^*) = \mathbf{0}$ *and* $\nabla^2\mathcal{J}(\mathbf{u}^*)$ *is positive definite. Then* $\mathbf{u}^*$ *is a strict local minimum of* $\mathcal{J}$*.*

Note that the second-order sufficient conditions are not necessary, since point $\mathbf{u}^*$ can be a strict local minimum, even though it fails to satisfy the sufficient conditions (an example is given in Nocedal et al. [305, p.17]).

Convexity of $\mathcal{J}$ provides some desirable and simplifying properties for analysis. These are described by the following theorem [33, 305].

Theorem 4.1.4. *When $\mathcal{J}$ is convex, then any local minimum $\mathbf{u}^*$ is also a global minimum of $\mathcal{J}$. If, in addition, $\mathcal{J}$ is differentiable, then any stationary point $\mathbf{u}^*$ is also a global minimum of $\mathcal{J}$. If $\mathcal{J}$ is strictly convex, then $\mathbf{u}^*$ is the unique global minimum of $\mathcal{J}$.*

Optimality conditions in non-smooth problems

If $\mathcal{J} \notin \mathbf{C}^1$, the aforementioned definitions are not necessarily valid anymore. In this section generalization of optimality conditions to non-smooth problems are presented based on [267]. In order to define the optimality conditions in non-smooth case, the concepts of the Lipschitz continuous function, generalized directional derivative, subdifferential, and subgradient are needed. The Lipschitz continuous function is defined as follows:

Definition 4.1.7. *A function $\mathcal{J} : \mathbb{R}^p \to \mathbb{R}$ is locally Lipschitz continuous at $\mathbf{u}^* \in \mathbb{R}^p$, if there exist Lipschitz-constant $K > 0$ and $\delta > 0$ such that*

$$|\mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{v})| \leq K\|\mathbf{u} - \mathbf{v}\|_2 \text{ for all } \mathbf{u}, \mathbf{v} \in B(u^*, \delta).$$

The concepts of subgradient and subdifferential generalize the principles of the ordinary differentiability of smooth and functions for non-smooth Lipschitz continuous functions. In the case of the Lipschitz continuous function, there does not necessarily exist classical directional derivatives. This means that the generalization of the ordinary directional derivatives to consider also non-smooth Lipschitz functions is needed and given by the following definition [267, p.29].

Definition 4.1.8. *Let $\mathcal{J} : \mathbb{R}^p \to \mathbb{R}$ be a locally Lipschitz continuous function at $\mathbf{u} \in \mathbb{R}^p$. The generalized directional derivative of $\mathcal{J}$ at $\mathbf{u}$ in the direction $\mathbf{d} \in \mathbb{R}^p$ is defined by*

$$\mathcal{J}^o(\mathbf{u}; \mathbf{d}) = \limsup_{\mathbf{v} \to \mathbf{u}, t \downarrow 0} \frac{\mathcal{J}(\mathbf{v} + t\mathbf{d}) - \mathcal{J}(\mathbf{v})}{t}.$$

The generalized directional derivative always exists for Lipschitz continuous functions and coincides with the ordinary directional derivative $\mathcal{J}'(\mathbf{u}; \mathbf{d})$ when it exists [267]. Using the generalized directional derivatives, one can define the concept of subdifferential and subgradient for Lipschitz functions.

Definition 4.1.9. *Let $\mathcal{J} : \mathbb{R}^p \to \mathbb{R}$ be locally Lipschitz continuous. The subdifferential $\partial\mathcal{J}$ (according to [74]) of $\mathcal{J}$ at $\mathbf{u} \in \mathbb{R}^p$ is defined by*

$$\partial\mathcal{J}(\mathbf{u}) = \{\boldsymbol{\xi} \in \mathbb{R}^p | \mathcal{J}^o(\mathbf{u}; \mathbf{d}) \geq \boldsymbol{\xi}^T\mathbf{d} \quad \text{for all} \quad \mathbf{d} \in \mathbb{R}^p\}.$$

Each element $\boldsymbol{\xi} \in \partial\mathcal{J}(\mathbf{u})$ is called a subgradient of $\mathcal{J}$ at $\mathbf{u}$.

More details about properties and calculus of the subdifferential and subgradient can be found in [267, 332].

Using the generalized directional derivatives and subdifferential, the classical first order optimality conditions can be generalized for unconstrained nonsmooth optimization. Hence, the necessary conditions for a Lipschitz function to attain its local minimum are given by the next theorem [267, p.70].

Theorem 4.1.5. *If $\mathcal{J}$ is locally Lipschitz at $\mathbf{u}^*$ and attains its local minimum at $\mathbf{u}^*$, then*
(i) $\mathbf{0} \in \partial\mathcal{J}(\mathbf{u}^*)$ *and*
(ii) $\mathcal{J}^o(\mathbf{u}^*; \mathbf{d}) \geq 0$ *for all* $\mathbf{d} \in \mathbb{R}^p$.

For convex functions the above conditions are also sufficient and the minimum is global. Finally, the concept of the stationary point can be generalized for a Lipschitz continuous function.

Definition 4.1.10. *Let $\mathcal{J}$ be locally Lipschitz continuous. Point $\mathbf{u}^* \in \mathbb{R}^p$ is called a substationary point of the minimization problem* (13)*, if*

$$\mathbf{0} \in \partial\mathcal{J}(\mathbf{u}^*).$$

4.2 Basic optimization algorithms

Perhaps the most usual approach to cope with non-smoothness is the use of nonderivative optimization such as simplex or coordinate descent methods [242]. Another choice is to smooth the original problem by a suitable function approximation and optimize it by using an appropriate gradient-based optimization method. Beginning at $\mathbf{u}^0 \in \mathbb{R}^p$, optimization algorithms generate a sequence of iterates $\{\mathbf{u}^t\}_{t=0}^{\infty}$ that terminates when one or more stopping criteria are satisfied. This means that no more progress can be made or the accuracy of the solution point is approximated sufficiently [305, p.19]. For developing and evaluating different solvers for the problem of the spatial median, a set of basic optimization methods are described in this section.

4.2.1 Gradient-based optimization methods

Smooth nonlinear optimization problems are usually solved by a method that utilizes derivative information. As described above, the derivative information can be used for determining the optimality conditions of the solution. If the optimality conditions are not yet satisfied, the best possible search direction is selected before the line search by using the gradient of the objective function.

Such methods are usually referred to as gradient-based methods. Minimization of objective function $\mathcal{J}$ with a gradient-based method is performed by iterating between two basic steps:

1. Find the best descent search direction $\mathbf{d} \in \mathbb{R}^p$.

2. Find the optimal step size λ in the direction of vector $\mathbf{d}$ such that the value of $\mathcal{J}$ becomes minimized in the search direction.

Numerous rules and algorithms following the aforementioned basic steps and gradient principle have been developed (see, e.g., [33]). The most naive way to utilize the derivative information of objective function $\mathcal{J}$ is to constantly progress to the direction of the steepest descent that is obtained by $-\nabla\mathcal{J}$ (the negative direction to the gradient of $\mathcal{J}$). This is the principle of the well-known *steepest descent method* (a.k.a. *gradient method*) [33, pp.25–26]. The steepest descent method is simple, but it often leads to slow convergence. The problem is that on an elongated cost surface, the gradient direction can be almost orthogonal to the direction that leads to the minimizing point of the objective function and makes the steepest descent algorithm to progress with short orthogonal "zigzag" steps [33, p.26].

Conjugate gradient method

Conjugate gradient method (CG) is a more advanced gradient-based approach to the nonlinear smooth optimization. CG methods were originally developed for solving large linear systems of equations (see, e.g., articles in [240]). The first nonlinear CG method was developed in the 1960s and many variants have been proposed since then [305]. CG methods improve the convergence speed of the steepest descent method by reusing the derivative information from the previous iterations. Another significant advantage is that no matrix storage is required. Because CG methods naturally assume well-defined gradients in the whole search space, non-smooth optimization problems can not be solved without approximations.

The nonlinear CG optimization algorithm is given next. For the convergence of the algorithm, it is assumed that level set $\mathcal{L} := \{\mathbf{u} : \mathcal{J}(\mathbf{u}) \leq \mathcal{J}(\mathbf{u}^0)\}$ is bounded. Moreover, in some neighborhood $\mathcal{N}$ of $\mathcal{L}$, the gradient of objective function $\mathcal{J}$ is assumed to be Lipschitz continuous (cf. Definition 4.1.7) [305, p.127].

Algorithm 4.2.1. CG algorithm

Step 1.(*Initialization.*) For a given $\mathbf{u}^0 \in \mathbb{R}^p$ evaluate $\mathcal{J}^0 = \mathcal{J}(\mathbf{u}^0)$ and $\nabla\mathcal{J}^0 = \nabla\mathcal{J}(\mathbf{u}^0)$. Set $\mathbf{d}^0 = -\nabla\mathcal{J}^0$ and $t = 0$.

Step 2.(*Line search.*) Solve α^t using a suitable line search method and set $\mathbf{u}^{t+1} = \mathbf{u}^t + \alpha^t\mathbf{d}^t$.

Step 3.(*Update of search direction.*) Evaluate $\nabla\mathcal{J}^{t+1}$. Set

$$\beta^{t+1} = \frac{\|\nabla\mathcal{J}^{t+1}\|_2^2}{\|\nabla\mathcal{J}^t\|_2^2} \tag{14}$$

or

$$\beta^{t+1} = \frac{(\nabla \mathcal{J}^{t+1})^T (\nabla \mathcal{J}^{t+1} - \nabla \mathcal{J}^t)}{\|\nabla \mathcal{J}^t\|_2^2} \tag{15}$$

and $\mathbf{d}^{t+1} = -\nabla \mathcal{J}^{t+1} + \beta^{t+1}\mathbf{d}^t$. Set $t = t + 1$.

Step 4. (*Stopping criterion.*) If satisfied then stop. Otherwise go to step 2.

Use of (14) leads to the *Fletcher-Reeves* and (15) to *Polak-Ribiere* CG method. The CG algorithm attains the global solution only for a strictly convex objective function.

4.2.2 Direct search methods

Also a number of non-derivative methods, such as the finite difference, coordinate descent, and direct search methods exist for non-smooth problems [33]. Since these methods do not utilize derivative information, they are inherently appropriate for solving non-smooth optimization problems. Direct search methods are also referred to as zero-order-methods ($C^0(\mathbb{R}^p)$), because instead of using derivative information, they construct approximations of the cost functions [251]. Margaret Wright [398] defines the class of direct search methods as follows:

- *A direct search method uses only function values.*
- *A direct search method does not "in its heart" develop an approximate gradient.*

Wright reminds us that the second criterion is ill-defined, since any comparison of function values can be considered as a development of an approximate gradient. Overall, direct search methods are widely used for unconstrained nonlinear optimization problems. Implementations can be found, for example, in the MATLAB optimization toolbox (see http://www.mathworks.com/) and [323].

Nelder-Mead Simplex Method

The *Nelder-Mead algorithm* (NM) is a heuristic direct search algorithm that was developed in the sixties by J.A. Nelder and R. Mead [300]. Because the algorithm is based on a geometric figure called *simplex*, it is also characterized as a *simplex method* that is a subclass of direct search methods. NM has been utilized, for example, in robust computer vision problems [278]. A Golden-Section search based variant of NM, referred to as NM-GS, is presented in [298]. NM-GS shows better theoretical convergence properties than the original NM, but since their practical performance was reported to be comparable to each other, the variant is skipped in this thesis.

The simplex is a geometric figure in a p-dimensional space that corresponds to a convex hull of $p + 1$ vertices with nonzero volume. For example, a simplex

in two or three dimensional spaces correspond to a triangle and tetrahedron, respectively. The simplex is maintained non-degenerated during each step of the algorithm.

There are not many theoretical results of the convergence of NM, but generally it is assumed to converge relatively fast in a close neighborhood of a global minimum, provided that the objective function $\mathcal{J}$ is strictly convex. Hence, if an approximate solution to an optimization problem can be given, then the exact solution can be found in a relatively short time by NM. On the other hand, when started with an arbitrarily chosen simplex, an optimal solution can not be guaranteed.

Because the derivative information is missing, the assessment of the local optimality conditions is more complicated. Naturally, the optimality conditions, given in Theorems 4.1.1, 4.1.2, 4.1.3, and 4.1.5, are not of use if derivative information is not available. This complicates the definition of the termination criteria for the algorithm. Another weakness of NM is its poor scalability to large-scale problems. Large-scale problems usually lead to vast numbers of objective function evaluations that significantly increase computational cost and, thereby, lengthen the time to converge.

Regardless of its popularity as a nonlinear optimization method, the literature indicates that the poor effectiveness and extreme sensitivity of the solutions to the stopping criteria make NM too slow and unreliable method for large DM applications. On the other hand, when initialized with an approximate gradient-based method, NM may converge fast to the locally optimal solution.

Algorithm

The NM algorithm creates a sequence of simplices with $p+1$ vertices in $\mathbb{R}^p$. A re-formed simplex at t^{th} iteration is denoted by $\{\mathbf{u}_1^t, \ldots, \mathbf{u}_{p+1}^t\}$.

The vertices are ordered at each iteration such that the condition $\mathcal{J}(\mathbf{u}_1^t) \leq \mathcal{J}(\mathbf{u}_2^t) \leq \ldots \leq \mathcal{J}(\mathbf{u}_{p+1}^t)$ is satisfied. Because iteration index t has no other meaning than to provide an alternative stopping criterion, that is the maximum number of iterations for the algorithm, it can be omitted for the rest of the algorithm description.

At each iteration, the vertices with the smallest (the best vertex $\mathbf{u}_1$) and the largest (the worst vertex $\mathbf{u}_{p+1}$) value of $\mathcal{J}$ are chosen. The worst point $\mathbf{u}_{p+1}$ is shifted to a new position $\mathbf{u} \in \mathbb{R}^p$ such that condition $\mathcal{J}(\mathbf{u}) < \mathcal{J}(\mathbf{u}_{p+1})$ becomes satisfied. $\mathbf{u}$ is determined by applying the following operations to the simplex: *reflection, expansion, contraction*, and *shrinkage*. The operations are associated with scale parameters that are denoted by ρ, χ, γ, and σ, respectively. The values of these parameters must satisfy the following conditions:

$$\rho \geq 0,\ \chi \geq 1,\ 0 \leq \gamma \leq 1,\ \text{and}\ 0 \leq \sigma \leq 1.$$

Typical parameter values can be [242] (cf., MATLAB Optimization Toolbox[1]):

$$\rho = 1,\ \chi = 2,\ \gamma = \frac{1}{2},\ \text{and}\ \sigma = \frac{1}{2}.$$

The outcome of each iteration is either an accepted point to replace the worst vertex $\mathbf{u}_{p+1}$ or, provided that the shrinkage operation was carried out, the outcome can be p new points that form a new simplex with the existing vertex $\mathbf{u}_1$ for the next iteration. The search direction of an operation is defined by the worst vertex $\mathbf{u}_{p+1}$ and $\bar{\mathbf{u}}$, which is the mean of all vertices except $\mathbf{u}_{p+1}$ and given by

$$\bar{\mathbf{u}} = \frac{1}{p}\sum_{i=1}^{p}\mathbf{u}_i.$$

The iteration of the NM algorithm is given next according to [242].

Algorithm 4.2.2. Nelder-Mead algorithm

Step 1. (*Order*) Sort the vertices of the current simplex to satisfy $\mathcal{J}(\mathbf{u}_1) \leq \mathcal{J}(\mathbf{u}_2) \leq \ldots \leq \mathcal{J}(\mathbf{u}_{p+1})$.

Step 2. (*Reflect*) Compute the *reflection point* $\mathbf{u}_r$ from

$$\mathbf{u}_r = \bar{\mathbf{u}} + \rho(\bar{\mathbf{u}} - \mathbf{u}_{p+1}). \tag{16}$$

Evaluate $\mathcal{J}_r = \mathcal{J}(\mathbf{u}_r)$.

Step 3. (*Expand*) If $\mathcal{J}_r < \mathcal{J}_1$, then calculate the *expansion point* $\mathbf{u}_e$,

$$\mathbf{u}_e = \bar{\mathbf{u}} + \chi(\mathbf{u}_r - \bar{\mathbf{u}}), \tag{17}$$

and evaluate $\mathcal{J}_e = \mathcal{J}(\mathbf{u}_e)$. If $\mathcal{J}_e < \mathcal{J}_r$, then accept $\mathbf{u}_e$ and terminate the iteration; otherwise accept $\mathbf{u}_r$ and terminate the iteration.

Step 4. (*Contract*) If $\mathcal{J}_r \geq \mathcal{J}_p$, then perform a *construction* between $\bar{\mathbf{u}}$ and the better of $\mathbf{u}_{p+1}$ and $\mathbf{u}_r$.

Step 4.a. (*Outside*) If $\mathcal{J}_p \leq \mathcal{J}_r < \mathcal{J}_{p+1}$ (i.e., $\mathbf{u}_r$ is strictly better than $\mathbf{u}_{p+1}$), then perform an *outside contraction*: calculate

$$\mathbf{u}_c = \bar{\mathbf{u}} + \gamma(\mathbf{u}_r - \bar{\mathbf{u}}) \tag{18}$$

and evaluate $\mathcal{J}_c = \mathcal{J}(\mathbf{u}_c)$. If $\mathcal{J}_c \leq \mathcal{J}_r$, then accept $\mathbf{u}_c$ and terminate the iteration; otherwise, go to Step 5 (perform a shrink).

Step 4.b. (*Inside*) If $\mathcal{J}_r \geq \mathcal{J}_{p+1}$, then perform an *inside contraction*: calculate

$$\mathbf{u}_{cc} = \bar{\mathbf{u}} + \gamma(\bar{\mathbf{u}} - \mathbf{u}_{p+1}) \tag{19}$$

and evaluate $\mathcal{J}_{cc} = \mathcal{J}(\mathbf{u}_{cc})$. If $\mathcal{J}_{cc} < \mathcal{J}_{p+1}$, then accept $\mathbf{u}_{cc}$ and terminate the iteration; otherwise, go to Step 5 (perform a shrink).

[1] http://www.mathworks.com

Step 5. (*Shrink*) Evaluate $\mathcal{J}$ at the p points $\mathbf{v}_i = \mathbf{u}_1 + \rho(\mathbf{u}_i - \mathbf{u}_1)$, $i = 2, \ldots, p+1$. The (unordered) vertices of the new simplex for the next iteration consist of points $\{\mathbf{u}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{p+1}\}$.

As a stopping criterion one can use, for example:

$$\max_{2 \leq i \leq p+1} \|\mathcal{J}_1 - \mathcal{J}_i\|_1 < \varepsilon \quad \text{or} \quad \max_{2 \leq i \leq p+1} \|\mathbf{u}_1 - \mathbf{u}_i\|_2 < \varepsilon,$$

where ε is a small positive real number. The criteria are very sensitive to the scale of the problem and the prior knowledge of the problem space helps in the determination of the appropriate values.

4.2.3 Successive overrelaxation method

The *successive overrelaxation method* (SOR) [384, 56] is an iterative method originally proposed for solving a linear system of equations

$$\mathbf{A}\mathbf{u} = \mathbf{b}, \tag{20}$$

where $\mathbf{A}$ is $p \times p$ real matrix and $\mathbf{b}$ $p \times 1$ vector. The solution vector $\mathbf{u}$ exists and is unique if and only if $\mathbf{A}$ is nonsingular, and this solution is given explicitly by

$$\mathbf{u} = \mathbf{A}^{-1}\mathbf{b}. \tag{21}$$

Iterative methods, such as the Jacobi, Gauss-Seidel, and SOR methods, can be used for solving a problem presented in (20) by constructing a sequence of solutions $\{\mathbf{u}^t\}_{t=0}^{\infty}$. The idea of SOR is to accelerate the convergence rate of Gauss-Seidel type of methods, which are based on solving coordinate-wisely

$$(\mathbf{u}^{t-1/2})_i = \frac{1}{a_{ii}} \left((\mathbf{b})_i - \sum_{j<i} a_{ij} (\mathbf{u}^{t-1/2})_j - \sum_{j>i} a_{ij} (\mathbf{u}^{t-1})_j \right), \quad i = 1, \ldots, p.$$

The acceleration is realized by the following convex combination

$$\mathbf{u}^t = \mathbf{u}^{t-1} + \omega(\mathbf{u}^{t-1/2} - \mathbf{u}^{t-1}) = \omega \mathbf{u}^{t-1/2} + (1-\omega)\mathbf{u}^{t-1}, \tag{22}$$

where ω is the *relaxation factor* (a.k.a overrelaxation parameter or *extrapolation factor* [384, 395]). ω should be chosen so that it accelerates the convergence rate of Gauss-Seidel type of iterations. Practically the best value of ω is unknown, but some heuristics have been developed. A theorem due to Kahan shows that SOR fails to converge if ω is outside the interval $]0, 2[$ [155].

Theorem 4.2.1 (Kahan[2])**.** *A necessary condition for the SOR method to converge is* $|\omega - 1| < 1$. *(For* $\omega \in \mathbb{R}$ *this condition becomes* $\omega \in (0, 2)$*.)*

Note that if $\omega = 1$ then the method simplifies to the Gauss-Seidel method. If $\omega < 1$, the term *under-relaxation* is used. The SOR method has been applied, for example, to massive discrimination problems with support vector machines [269].

[2] Original reference not available to the authors: Kahan, W., Gauss-Seidel methods of solving large systems of linear equations, Doctoral Thesis, University of Toronto, Canada, 1958.

4.3 Classical statistical estimation

In this section, the most important statistical definitions related to the statistical estimation are given for later statistical analyzes and experiments.

4.3.1 Basic terminology

Many times in practical situations, it is not possible to gather measurements from the entire target population. Therefore, the estimation of the statistical parameter values is usually based on samples that are drawn from the population. To accomplish such tasks, the following definitions are essential to know [395]:

An estimate denoted by $\tilde{\boldsymbol{\theta}}$, is an educated, and hopefully 'the best', guess, that is based on known information, for the true value of a population parameter. Often, an estimate for the uncertainty of an estimate $\tilde{\boldsymbol{\theta}}$ can also be determined statistically.

An estimator defined by a functional T is a rule that tells how to calculate the estimate $\tilde{\boldsymbol{\theta}}$ based on the measurements contained in a sample.

It is worthwhile to note that when all measurements from a population are available, one should rather talk about calculation of a parameter value than its estimation. On the other hand, an estimate is also a random (vector) variable that has a distribution. The different statistical measures can be used for selecting the best estimator, whose "goodness" depends on the underlying distribution of the target sample. If the underlying statistical conditions are unknown, then the estimation is a more difficult task.

A location parameter determines the true position of the population distribution. It is defined as follows ([188]):

Definition 4.3.1. *Let $f(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\lambda})$ be the density function of a random vector variable $\mathbf{x}$. $\boldsymbol{\theta}$ is a location parameter if the density $f(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\lambda})$ can be written as a function of $\mathbf{x}-\boldsymbol{\theta}$; that is, $f(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\lambda}) = h(\mathbf{x}-\boldsymbol{\theta};\boldsymbol{\lambda})$ for some function $h(\cdot;\boldsymbol{\lambda})$, and $h(\cdot;\boldsymbol{\lambda})$ does not depend on $\boldsymbol{\theta}$.*

The location parameter of an interesting population is estimated by using a *point estimator*, whose outcome is naturally called *point estimate*. It is the actual numerical value that approximates the unknown and exact value of the location parameter on a given sample. Instead of a point estimate, a scatter estimate of a given sample may also be of interest. Such an approach is referred to as *interval estimation*, which is another type of statistical estimation.

The following definitions are mainly taken from [135]. Let $\boldsymbol{\Theta} \subset \mathbb{R}^p$ be an arbitrary parameter space. $\mathbf{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$ is a random sample in $\mathbb{R}^p$, which is drawn from a population with a probability density function $p(\mathbf{x},\boldsymbol{\theta})$, where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. $\mathbf{X}$ is assumed to be independent and identically distributed. The estimate

of the true parameter vector $\boldsymbol{\theta}$ is denoted by $\tilde{\boldsymbol{\theta}}$, which is a function of T over the random sample $\mathbf{X}$ as

$$\tilde{\boldsymbol{\theta}} = T(\mathbf{x}_1, \ldots, \mathbf{x}_n) = T(\mathbf{X}).$$

Because $\mathbf{X}$ consists of random vectors $\mathbf{x}_i$, also $\tilde{\boldsymbol{\theta}}$ becomes a random vector. The *expected vector* or the *mean* of a random sample $\mathbf{X}$ is defined by

$$\mathcal{E}\{\mathbf{X}\} = \sum_{i=1}^{n} \mathbf{x}_i p(\mathbf{x}_i),$$

where $p(\mathbf{x})$ is the joint density function of $\mathbf{x}$. The sample mean vector is defined by

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i.$$

The *variance* of an individual random variable $x_j = (\mathbf{x}_i)_j$ ($i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, p\}$) is defined by

$$\sigma_j = \mathcal{E}\{(x_j - \mu_j)(x_j - \mu_j)\},$$

where $\mu_j = (\boldsymbol{\mu})_j$. The unbiased estimate of *covariance matrix* $\boldsymbol{\Sigma}$ of a random sample $\mathbf{X}$ is defined by

$$\tilde{\boldsymbol{\Sigma}} = \mathcal{E}\{(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T\} = \frac{1}{n-1} \sum_{1}^{n} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T.$$

The bias of an estimate $\tilde{\boldsymbol{\theta}} = T(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the difference between the true value of the estimated parameter $\boldsymbol{\theta}$ and the expected value of $\tilde{\boldsymbol{\theta}}$. According to the classic statistics, the best estimator one can find should be unbiased, consistent, and efficient [135, p.124]. Hence, the definitions of these central properties are presented next.

Definition 4.3.2. *An estimate $\tilde{\boldsymbol{\theta}} = T(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is said to be an unbiased estimate of $\boldsymbol{\theta}$ if*

$$\textit{bias} = \mathcal{E}\{\tilde{\boldsymbol{\theta}}\} - \boldsymbol{\theta} = 0 \quad \textit{for all} \quad \boldsymbol{\theta} \in \boldsymbol{\Theta}.$$

Otherwise, it is a biased estimate.

For example, the sample mean is an unbiased estimator of the population mean. In practice, however, the bias is usually impossible to determine, because the exact value of the true parameter $\boldsymbol{\theta}$ tends to be unknown. The consistency is defined as:

Definition 4.3.3. *An estimate* $\tilde{\boldsymbol{\theta}} = T(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ *is a consistent estimate of* $\boldsymbol{\theta}$ *if and only if*

$$\lim_{n\to\infty} \mathcal{E}\{\|T(\mathbf{x}_1, \ldots, \mathbf{x}_n) - \boldsymbol{\theta}\|_2\} = 0.$$

A consistent estimator guarantees asymptotically correct estimates. The efficiency of an estimator is measured by the variance of the estimates. The relative efficiency of two consistent estimators tells which one gives the correct values in probability.

Definition 4.3.4. *Let* $\tilde{\boldsymbol{\theta}}_1 = T_1(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ *and* $\tilde{\boldsymbol{\theta}}_2 = T_2(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ *be the estimators for the true parameter* $\boldsymbol{\theta}$*. When the estimates* $\tilde{\boldsymbol{\theta}}_1$ *and* $\tilde{\boldsymbol{\theta}}_2$ *are compared on the same samples, their relative efficiency is defined by the ratio*

$$\eta = \frac{\mathcal{E}\{\|\tilde{\boldsymbol{\theta}}_1 - \boldsymbol{\theta}\|^2\}}{\mathcal{E}\{\|\tilde{\boldsymbol{\theta}}_2 - \boldsymbol{\theta}\|^2\}}.$$

If $\eta \leq 1$ always for all possible $\tilde{\boldsymbol{\theta}}_2$, then $\tilde{\boldsymbol{\theta}}_1$ is said to be an efficient estimate. The efficiency of an estimator depends on the underlying statistical distribution. The sample mean is the most efficient estimator on the samples that are drawn from the normal distribution. For a nonsymmetric (skewed) distribution this does not necessarily hold anymore and, for example, the median may be the more efficient estimator.

All estimators proposed in this study are expected to satisfy the Fisher consistency. A Fisher consistent estimator measures the right quantity at the idealized parametric model [162].

Definition 4.3.5. *(Hampel [162]). Let* $F_{\boldsymbol{\theta}}$ *be a family of probability density functions* $f(\mathbf{x}, \boldsymbol{\theta})$*. Let* $\{F_{\boldsymbol{\theta}}, \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ *denote a parametric model, where* $\boldsymbol{\Theta}$*, the* $F_{\boldsymbol{\theta}}$*'s and the mapping* $\boldsymbol{\theta} \to F_{\boldsymbol{\theta}}$ *are precise. Estimator* T *with values in* $\boldsymbol{\Theta}$ *is said to be Fisher consistent at the parametric model if and only if*

$$T(F_{\boldsymbol{\theta}}) = \boldsymbol{\theta} \quad \textit{for all} \quad \boldsymbol{\theta} \in \boldsymbol{\Theta}.$$

These are the basic properties that are expected to hold for all appropriate estimators. Another important factor considering the evaluation of an estimator is, of course, the computational cost required to compute the estimate.

4.3.2 Multidimensional transformations

Following the presentation in [338], a couple of definitions that are closely related to the multivariate location estimation problems are given. These definitions are needed for comparing the geometrical properties of the multivariate estimators.

Definition 4.3.6. *The multivariate location estimator is said to be translation (or location) equivariant if*

$$T(\mathbf{x}_1 + \mathbf{b}, \ldots, \mathbf{x}_n + \mathbf{b}) = T(\mathbf{x}_1, \ldots, \mathbf{x}_n) + \mathbf{b} \quad \textit{for any} \quad \mathbf{b} \in \mathbb{R}^n.$$

Definition 4.3.7. *Let π denote any permutation on $\{1, ..., n\}$. The multivariate location estimator is said to be permutation invariant if*

$$T(\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(n)}) = T(\mathbf{x}_1, \ldots, \mathbf{x}_n) \quad \textit{for any} \quad \pi.$$

Definition 4.3.8. *The multivariate location estimator is said to be affine equivariant (coordinate-free [89]) if*

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

where $\mathbf{b} \in \mathbb{R}^n$ and $\mathbf{A}$ is any nonsingular matrix.

Affine equivariance is a desirable property, but it is difficult to combine with robustness. Affine equivariant robust estimators need also more computation time than classical estimators [338, p.270].

4.4 From classical to robust statistics

Statistical inferences are based on both observations and prior assumptions about underlying conditions. The underlying conditions determine, for example, distributional models, randomness, independence, etc. The classical statistics rely on the normal (or Gaussian) theory that emerged due to finding that the errors in the least square (LS) problems are normally distributed. The LS problem is defined by the following optimization problem

$$\min_{\tilde{\boldsymbol{\theta}} \in \mathbb{R}^p} \mathcal{J}(\tilde{\boldsymbol{\theta}}), \quad \text{for} \quad \mathcal{J}(\tilde{\boldsymbol{\theta}}) = \sum_{i=1}^{n} \|\mathbf{x}_i - \tilde{\boldsymbol{\theta}}\|_2^2, \tag{23}$$

where $\tilde{\boldsymbol{\theta}}$ is an unknown parameter vector. The Euclidean distance is used between the observed values $\mathbf{x}_i$ and model parameter $\tilde{\boldsymbol{\theta}}$. Due to its explicitly solvable formulation LS problem is very fast to compute (a closed-form solution exists). This was of great importance at the time of its invention, since there were no computers available. Because of its simplicity, computational efficiency and perhaps also tradition, a large number of the techniques in statistical data mining and data analysis software packages are still based on the LS principle.

Real world data rarely satisfy the classical normal assumption. According to Huber [197], it has been clear since the sixties that one seldom has precise knowledge about the true underlying distributional conditions. John W. Tukey has been considered as the first one who in the sixties recognized and elaborated the problem of the extreme sensitivity of classical procedures to very minor deviations from the normal assumptions [199].

4.4.1 Robustness

According to Huber [195] "*robustness signifies insensitivity to small deviations from the assumptions*". A small deviation from the assumptions refers either to gross errors in a minor part of the data or small errors in a large part of the data. The primary goal of the robust procedures is to safeguard against those errors.

A typical deviant in a data set is an outlying data value. An observation with one or more such data values deviates significantly from the bulk of the data and is therefore called an *outlier* [24]. There are many reasons for the existence of such observations in real-world data sets. They can be caused by a failure in a data acquisition system or by a human mistake. On the other hand, an outlying value may also be a correct measurement of an object with deviating features.

Another kind of deviation from the normal assumptions is missing data (see Section 3.2.8). There can be many reasons for missing data values (see, e.g., [258]). Fortunately, if the mechanism that leads to the missing values is known, it may be possible to substitute these missing values by the correct or estimated values. However, in the case of sparse and large data sets, it may be too laborious to analyze and substitute the missing values. Moreover, the missing data mechanism may be fully unknown, which makes the estimation of the correct value difficult. The same holds for the outliers. For instance, missing beats and extra beats in heart rate time series may arise either due to a physiological reason or measurement errors [339].

These facts have been one of the most significant motivation for the development of robust procedures. Regardless, statistical tools are still mainly based on the classical statistics. This prevents precise and correct inferences from dirty real-life data, even though the actual algorithms are fully reliable.

A competing factor for robustness is (statistical) efficiency. Usually the increased robustness leads to decreased efficiency. For example, the trimmed sample mean estimator is less efficient than the sample mean computed for the whole data set. The trimmed variant loses a part of the data for diminishing the influence of gross-errors. The less data the less information and, finally, the more uncertainty. In practice, this is shown as increased variance in the estimates. For this reason, Coakley et al. [75] propose that the relative efficiency of a robust estimator should provide at least 95% efficiency with respect to the least squared estimators on a normally distributed sample.

From the point of view of the software developers, especially the ones working with DM and KDD methods and algorithms, the robust estimators should also have efficient implementations with respect to the amount of required computation and memory usage, testability, etc. (computational efficiency should not be confused with statistical efficiency). On the other hand, approximation algorithms that are implemented in order to shorten computation time have been considered as a risk for the consistency and breakdown point of the estimators [179]. Therefore, it may be worthwhile to test new methods both from the computational and statistical point of view.

4.4.2 Outlier trimming

How the aforementioned deviations should be taken into account in the operations on incomplete and erroneous real-world data sets? The simplest approach is to completely reject outliers and apply classical statistics to the remaining data. One may use robust statistics (robust location and scatter estimates) in two ways. Either one uses the robust estimates instead of the classic statistics or, first computes a robust estimate in order to identify outliers, then rejects or corrects those, and afterwards applies classic methods to the cleaned data.

Trimming is an example of the rejection techniques that is used to make the classical estimators more robust. Trimming means that a predetermined fraction or number of the most extreme observations are removed from the sample before the estimate is computed (e.g., [24]). *The trimmed mean* is a trimmed variant of the sample mean.

An effort to define the correct trimming fractions was made by Stigler [360], who examined eleven location estimators on 24 real world data sets. The result was that these data sets contain such minor deviations from the normal assumptions that a very fine trimming (10%) yields the best estimates and, moreover, the second best in the tests was the sample mean without trimming. Huber comments on Stigler's aforementioned real-world experiments by stating that the used real-world data sets contained fewer gross errors than the average real world data (see discussions in [360]).

Hence, the trimming approach is not as simple as it looks. The trimming procedure is sensitive to the changes at the rejection point [196]. A high density of data at these points may distort the estimate seriously. Another problem is to define the trimming fraction or limit. Outlier detection from multivariate data is not easy. Visual recognition of outliers from high dimensional data sets is difficult, which often makes the explorative methods useless. A further complicating fact is the difficulty to give a reliable characterization for extremity, because it depends on the unknown location parameter of the sample. Hence, this leads to a kind of recursive problem, which proves that the separation of the rejection and estimation steps is not at all obvious [195, 197]. Due to false rejections and false retentions, even a normal data set containing few gross errors may not be normal after trimming. The situation is even worse when classical estimates are applied after trimming a data set that is mistakenly assumed normal for the main part. According to Huber [195], the best robust procedures outperform rejection procedures, because they are based on smooth transition from the full acceptance to the full rejection of observations.

4.4.3 Quantification of robustness

In order to be able to analyze and compare the robustness of estimators, measures of robustness are needed. Two main types of robustness exist: qualitative and quantitative.

Qualitative robustness

Qualitative robustness is based on continuity of the estimators [195, 162]. In the qualitative sense, the robust estimators possess a so-called weak-star continuity: if two empirical cumulative distributions get closer to each other, then the estimates based on these samples also get closer to each other [162]. The exact mathematical formulation behind the weak-star topology is skipped here (see more details, e.g., in [195]).

Quantitative robustness

Quantitative robustness expresses the effect of small deviations from the underlying distributional conditions to the distribution of an estimator [195, 163]. From the perspective of this thesis, this type of robustness is of main interest.

Breakdown point Perhaps the most popular of the quantitative measures is the so-called *breakdown point* (BP). Rousseeuw and Leroy [338] define BP as *"the smallest fraction of contamination that can cause the estimator T to take values arbitrarily far from $T(X)$"*. A formal definition according to [195, 338] follows next.

Definition 4.4.1. *Let $T = T(\mathbf{X})$ be any arbitrary estimator. Let us denote by $\mathbf{X}'$ all contaminated samples obtained by replacing m data points by arbitrary values (this allows also extremely distant outliers). The maximum bias caused by the contamination is given by:*

$$bias_{\max}(m, T, \mathbf{X}) = \sup_{\mathbf{X}'} \|T(\mathbf{X}') - T(\mathbf{X})\|, \tag{24}$$

where the supremum is taken over all contaminated sets $\mathbf{X}'$. The finite-sample breakdown point is given by

$$\varepsilon_n^*(T, \mathbf{X}) = \inf\{\frac{m}{n} : bias_{\max}(m, T, \mathbf{X}) = \infty\}. \tag{25}$$

Hence, BP measures the global reliability of an estimator. The above definition is independent on the probability distributions. In the case of a non-robust estimator, such as the sample mean, BP is zero [198]. On the other hand, the highest possible breakdown point for any translation equivariant estimator is 0.5 (cf. Theorem 4.4.1).

Theorem 4.4.1. *(Lopuhaä et al. [261]). Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a sample of n points in $\mathbb{R}^p$. When T_n is translation equivariant, then $\varepsilon^*(T_n, \mathbf{X}) \leq \lfloor (n+1)/2 \rfloor / n$, where $\lfloor u \rfloor$ denotes the nearest integer less than or equal to u.*

The upper limit is due to the self-evident truth that if more than half of the data is contaminated, it becomes impossible to decide which part of that data is good.

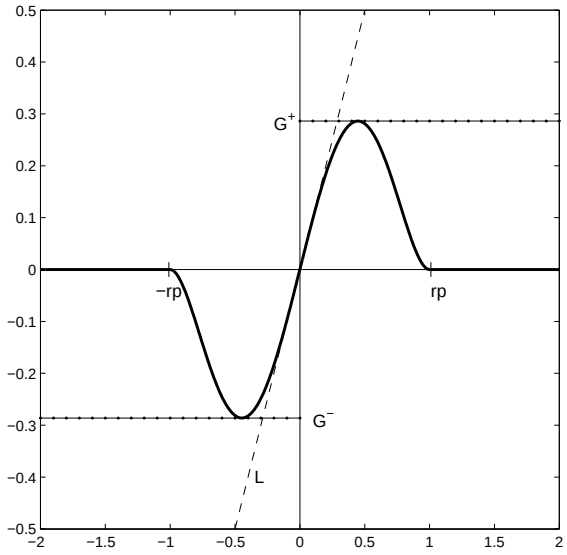


FIGURE 11 The influence curve of the Tukey's biweight estimator. $\max(G^+, G^-)$ is the gross-error sensitivity. L is the local-shift sensitivity and rp the rejection point.

Influence function and its derivations *The influence function* (IF) and its derivatives represent the local concepts of the robustness. They give an infinitesimal aspect to the robustness, since they describe the standardized effect of an infinitesimal contamination on the estimate [162, 24]. In a mathematical sense, the influence function is the first derivative of an estimator defined by a functional T. The value of the derivative at a given point $\mathbf{x}$ (when it exists, and is unique) measures the normalized effect by small contamination to the estimate at this point. Let's consider the asymptotic definition of the influence function first [163].

Definition 4.4.2. *Let us consider the functional form of an estimator $T = T(\mathbf{X})$ and an underlying basic distribution F. A random contaminated model with a contamination ratio λ is given by $(1-\lambda)F + \lambda G$. The influence function of a functional T at a distribution F is then given by*

$$IF(\xi, T, F) = \lim_{\lambda \to 0} \frac{T((1-\lambda)F + \lambda G) - T(F)}{\lambda}, \tag{26}$$

where G is the atomic distribution for which $P(X = \xi) = 1$.

The influence function is a useful measure of robustness as long as the contaminated fraction of the data is smaller than the breakdown point. For a robust estimator, the influence function should be bounded and continuous. The bounded influence function provides safety against outliers by determining an upper limit for the worst approximate effect that a fixed amount of contamination may have on the estimate. For example, both coordinate-wise and spatial medians have bounded influence functions. The continuity property provides safety against inliers, which means that not any single observation can determine the value of an estimator. For example, the coordinate-wise medians are deter-

mined by the location of one or two middle values [55], which makes them very sensitive to rounding and grouping of these points.

As data mining tasks usually suffer from a lack of knowledge about the underlying distribution, the finite-sample version of the influence function is also needed [163]. The empirical influence function is used when the intention is to measure the influence of a single data point to a particular estimate.

Definition 4.4.3. *Let us suppose that we have an estimator T_n for $n \geq 1$ and a sample $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$ of $n-1$ observations. The empirical influence function of T_n is a plot of*

$$T_n(\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}, \mathbf{x}) \tag{27}$$

as a function of $\mathbf{x}$.

Another finite-sample variant for the Hampel's influence function is the *sensitivity curve* that was proposed by Tukey around 1970 [199, 195]. It is actually also a variant of the Jackknife method [282]. The idea is to assess the influence of one additional (virtual) observation $\mathbf{x}$ on estimator $T_n = T(\mathbf{x}_1, \ldots, \mathbf{x}_n)$. As a function of an additional observation $\mathbf{x}$ that is scaled by the sample size n, the sensitivity curve is obtained from (26) by replacing F by F_{n-1} and λ by $1/n$ as

$$SC_{n-1}(\mathbf{x}) = n[T_n(\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}, \mathbf{x}) - T_{n-1}(\mathbf{x}_1, \ldots, \mathbf{x}_{n-1})]. \tag{28}$$

The finite gross-error sensitivity, which corresponds to the boundedness of the influence function, is defined as:

Definition 4.4.4. *The gross-error sensitivity [163, 162] of a consistent estimator T_n of T at distribution F reads as*

$$GES = \sup_{\mathbf{x}} \|IF(\mathbf{x}, T, F)\|. \tag{29}$$

The gross error sensitivity determines the worst approximate effect that a fixed amount of contamination may have on the estimate. If $GES \rightarrow \infty$ for a particular estimator, then the estimator is completely intolerant against outliers. The robust estimators, such as the sample median, Tukey's biweight, Andrews' wave, etc. usually possess the finite gross-error sensitivity. As an example, the gross-error sensitivity of the Tukey's biweight estimator is depicted in Figure 11.

The local-shift sensitivity expresses the worst approximate effect caused by removing an observation and reintroducing it at a new position. It is also based on the influence function and defined as follows:

Definition 4.4.5. *The local-shift sensitivity (e.g., [188, 162]) is the supremum of the absolute slopes of chords joining all pairs of distinct points on the influence function:*

$$LSS = \sup_{\mathbf{x} \neq \mathbf{y}} \frac{|IF(\mathbf{x}) - IF(\mathbf{y})|}{\|\mathbf{x} - \mathbf{y}\|}. \tag{30}$$

This is a measure of the local effect of rounding or grouping to the value of an observation. A discontinuous influence function leads to the infinite local-shift sensitivity. *LSS* of the coordinate-wise sample median is asymptotically infinite at the central part of the sensitivity curve. In practical applications this is reflected by the fact that one or two middle observations dominate the value of the estimate.

The rejection point expresses the largest distance after which the observations become rejected. In this region the influence function equals zero as the observations do not have any influence on the estimate. The rejection point is defined as [162]

Definition 4.4.6.

$$RP = \inf\{r > 0; IF(\mathbf{x}; T, F) = 0 \quad when \quad \|\mathbf{x}\| > r\}. \tag{31}$$

All observations that exceed *RP* are rejected by the estimator. A class of estimators with a finite rejection point is referred to as redescending estimators (see, e.g., [162, 164, 417]). These types of estimators are particularly well protected against sufficiently large outliers.

In addition to the classic measures, such as consistency and efficiency, robust estimators should also be provided with a high breakdown point, which means qualitative robustness. These are the desirable properties for all robust estimators [195, 410, 75]. Considerable values in the aforementioned measures assure that an estimator tolerates well small deviations from the assumed model and avoid "catastrophe" in the case of larger deviations. From the DM and data clustering point of view, it is important to have methods and estimators that are insensitive to gross errors, because such errors are difficult to detect from large multidimensional real-world data sets. In conclusion, a set of requirements for robust estimators collected from [162, 333] are the following:

High efficiency Nearly equal efficiency with maximum likelihood estimators under ideal parametric models.

Qualitative robustness Estimates are influenced just slightly by small deviations from the assumed model.

Quantitative robustness Estimates are protected against large amounts of contamination or single gross errors (high breakdown point).

Local-shift sensitivity Smooth reaction to rounding and grouping.

Rejection point Separation between outliers and the bulk of data.

Fisher consistency Estimation of right quantity (for parametric models).

Affine equivariance The solution should be independent of the scales of the variables (multivariate cases)

Computational practicality The solution should be obtainable in a practical amount of computing time, even in a high dimension and/or with large amounts of data.

4.5 M-estimation

M-estimators (a.k.a. maximum likelihood type estimators) are a specific class of statistical functionals based on generalization of the traditional least squares principle (see, e.g., [196, 197, 274, 195, 163]). The idea of maximum likelihood estimation (ML-estimation) is to find a parameter that maximizes (or minimizes) the likelihood function (or its logarithm). Let us give a definition for the general M-estimator [195, 163].

Definition 4.5.1. *Any estimator $T_n = T_n(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, whose value is defined by a minimizing point of a problem of the form*

$$\arg\min_{T_n} \sum_{i=1}^{n} \rho(\mathbf{x}_i; T_n), \tag{32}$$

or by an implicit equation

$$\sum_{i=1}^{n} \psi(\mathbf{x}_i; T_n) = 0, \tag{33}$$

where ρ is an arbitrary function, $\psi(\mathbf{x}_i, \boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \rho(\mathbf{x}_i, \boldsymbol{\theta})$, (when $\nabla_{\boldsymbol{\theta}}$ exists and is unique) is said to be an M-estimator (or a maximum likelihood type estimator).

The generalized multivariate M-estimator of a location parameter is defined as

$$T_n = \arg\min_{T_n} \sum_{i=1}^{n} \rho(\mathbf{x}_i - T_n), \tag{34}$$

or

$$\sum_{i=1}^{n} \psi(\mathbf{x}_i - T_n) = 0. \tag{35}$$

By altering ρ a large assortment of M-estimators for a location parameter can be constructed [196]. Usually ρ is strictly convex so that ψ becomes strictly monotone and consequently estimator T_n is unique.

By choosing $\rho(\mathbf{u}) = \mathbf{u}^2$, $\rho(\mathbf{u}) = \|\mathbf{u}\|_1$ and $\rho(\mathbf{u}) = -\log f(\mathbf{u})$ the multivariate sample mean, coordinate-wise median and maximum likelihood estimator, respectively, are obtained. $f(\mathbf{u})$ is the density function of a distribution. Note that the coordinatewise median is unique only if the number of observations is odd [213]. ψ is actually the influence function of the M-estimator. Hence, the sensitivity of the estimates can be evaluated from the shape of ψ. An estimator with small values of $\|\psi(\mathbf{u})\|_1$ on any large $\|u\|_1$ is robust against extreme observations. In Figure 13, the shape of the influence functions are depicted for four different

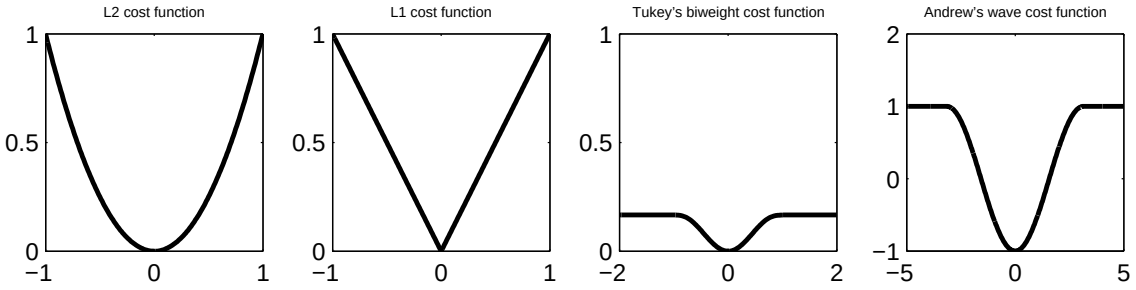


FIGURE 12 Cost functions of four M-estimators.

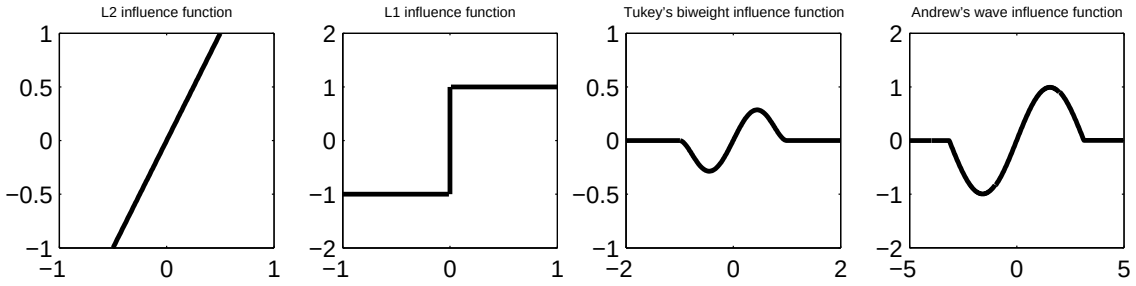


FIGURE 13 Influence functions of four M-estimators.

M-estimators. One can see that the l_2-estimator has an unbounded but strictly monotone influence curve, which expresses that already one outlier may totally disturb the estimate. A robust influence curve is given by the l_1-estimator that has a bounded and monotone influence function.

Tukey's biweight and Andrew's wave are redescending M-estimators [162, 164, 417]. The shape of the cost and influence functions are presented in Figures 12 and 13. Because the value of the influence functions equals zero for the estimator after the predetermined distance parameter (rejection point), all extreme outliers are entirely rejected. These estimators suffer from two drawbacks. First, they are not based on convex cost functions (i.e., monotone influence functions), which make finding of a globally optimal solution, which by Theorem 4.1.4 can not be assured to be unique, more complex. Secondly, the determination of the rejection limit is a difficult data-dependent problem.

A number of functions for M-estimators are introduced in [420]. When compared to other types of estimators, such as L- and R-estimators, M-estimators are more flexible and easily generalized to multivariate problems [163]. However, M-estimators are not inherently scale invariant (except the coordinate-wise median). Hence, in practical applications specific scale estimation procedures are needed. In this thesis, multivariate generalizations with missing data treatment are presented for three M-estimators.

4.6 Robust multivariate M-estimators and non-smooth optimization problem

In this section a pair of multivariate location estimators, that are based on the usual l_q-norms, are introduced along with a critical study from the point of view of robust statistics and optimization. By following the presentation by Kärkkäinen and Heikkola [213], multivariate formulations for the coordinate-wise median and the spatial median as a non-smooth optimization problem are given.

Let us consider the following family of optimization problems [213]:

$$\min_{\mathbf{u}\in\mathbb{R}^p} \mathcal{J}_q^{\alpha}(\mathbf{u}), \quad \text{for } \mathcal{J}_q^{\alpha}(\mathbf{u}) = \frac{1}{\alpha}\sum_{i=1}^{n} \|\mathbf{u}-\mathbf{x}_i\|_q^{\alpha}. \tag{36}$$

4.6.1 Coordinate-wise median

The marginal (or sample) median is a non-parametric estimate of the univariate population median. The p-variate median estimator can be simply constructed as a vector of marginal (sample) medians [338, 23]. The M-estimate formulation can be derived for the same estimator from the general M-estimation problem (32) [213].

In order to turn the multivariate coordinate-wise median to the M-estimation problem, we choose $q = \alpha = 1$ in (36) that leads to the minimization of the sum of l_1-norms. This is actually a non-smooth optimization problem and the subdifferential of the cost function is given by:

$$\partial\mathcal{J}_1^1(\mathbf{u}) = \sum_{i=1}^{n} \xi_i \quad \text{where}(\xi_i)_j = \text{sign}((\mathbf{u}-\mathbf{x}_i)_j). \tag{37}$$

The sign-function $\text{sign}(u)$ is defined such that $\text{sign}(u) = -1$ for $u < 0$, $sign(u) = 1$ for $u > 0$ and $sign(u) = [-1, 1]$ for $u = 0$. The solution of the problem is the coordinate-wise median. In practice, the solution represents the set of coordinate-wise middle values taken from the ordered sample set. The solution is unique for odd n, but for even n, all points in the closed interval between the middle values satisfy (37). An appropriate choice (used, e.g., in MATLAB) is the average of the two middle values.

4.6.2 Spatial median

The spatial median is a multivariate M-estimator of a location parameter. It is known by several names, such as Fermat-Weber point, multivariate L_1 estimator/median, the mediancentre or Weber point [352, 261, 338, 82, 148]. In operations research and management science it is best-known as Fermat-Weber point that minimizes the (weighted) sum of the Euclidean distances to the n given points in $\mathbb{R}^p$. Hence, it provides a solution for the usual facility location problem (fire stations, distribution or communication center, etc.) [262].

By choosing $q = 2\alpha = 2$ the problem of the spatial median is obtained from (36):

$$\min_{\mathbf{u}\in\mathbb{R}^p} \mathcal{J}_2^1(\mathbf{u}), \quad \text{for } \mathcal{J}_2^1(\mathbf{u}) = \sum_{i=1}^{n} \|\mathbf{u} - \mathbf{x}_i\|_2, \tag{38}$$

which clearly satisfies the conditions given in Definition 4.5.1. In the univariate case, (38) coincides with the coordinate-wise median.

The gradient of the convex cost function $\mathbf{f}(\mathbf{u}, \mathbf{x}_i) = \|\mathbf{u} - \mathbf{x}_i\|_2$ is well-defined and unique for all $\mathbf{u} \neq \mathbf{x}_i$. However, case $\mathbf{u} = \mathbf{x}_i$ leads to the use of the sub-gradient (see Definition 4.1.9), which is below characterized by the condition $\|\xi\|_2 \leq 1$ (see also [55]). Thus, the (local) extremity of (38) is characterized by means of a sub-gradient, which reads as

$$\partial\mathcal{J}_2^1(\mathbf{u}) = \sum_{i=1}^{n} \xi_i, \quad \text{with } \begin{cases} (\xi_i)_j = \frac{(\mathbf{u}-\mathbf{x}_i)_j}{\|\mathbf{u}-\mathbf{x}_i\|_2}, & \text{for } \|\mathbf{u} - \mathbf{x}_i\|_2 \neq 0, \\ \|\xi_i\|_2 \leq 1, & \text{for } \|\mathbf{u} - \mathbf{x}_i\|_2 = 0. \end{cases} \tag{39}$$

As pointed out in [213], (38) is a non-smooth optimization problem [267], which means that it can not be treated (both analytically and numerically) by using the classical (C^1) differential calculus.

Proof of existence and uniqueness of spatial median

When the inherent non-smoothness of problem (38) has been recognized, the existence and uniqueness of its solution need to be shown without using the mathematically incorrect characterization $\nabla\mathcal{J}_2^1(\mathbf{u}) = 0$. The existence of the minimizing solution for (38) is shown by proving the following theorem:

Theorem 4.6.1. *Problem* (38) *attains its minimum.*

Proof. Let us define a compact set

$$L = \bigcup_{i=1}^{n} B_c(\mathbf{x}_i, d), \text{where} \quad d = \max_{i,j\in\{1,\ldots,n\}} \|\mathbf{x}_i - \mathbf{x}_j\|_2.$$

It is clear that $\mathbf{x}_j \in B_c(\mathbf{x}_i, d)$ for all $i, j \in \{1, \ldots, n\}$ (this means that the intersection of B_c's contains all $\mathbf{x}_i$'s) and for any $\mathbf{u} \in L$ there exists at least one $\mathbf{x}_i$ such that $\|\mathbf{u} - \mathbf{x}_i\|_2 \leq d$. The complement of L is given by $M = \mathbb{R}^p \setminus L$. Because $\mathbf{u} \in M$ can not be inside any $B_c(\mathbf{x}_i, d)$, it follows that for any $\mathbf{u} \in M$ it holds $\|\mathbf{u} - \mathbf{x}_i\|_2 > d$.

1. Let $\mathbf{u} \in M$. Then $\|\mathbf{u} - \mathbf{x}_i\|_2 > d$ for all $\mathbf{x}_i$. From this it follows that $\sum_{i=1}^{n} \|\mathbf{u} - \mathbf{x}_i\|_2 > nd$ when $\mathbf{u} \in M$.

2. Now we have to show that there exists $\mathbf{u} \in L$ such that $\sum_{i=1}^{n} \|\mathbf{u} - \mathbf{x}_i\|_2 \leq nd$. However, this is trivial since by choosing $\mathbf{u} = \mathbf{x}_i$ for any $i = 1, \ldots, n$, we have $\sum_{i=1}^{n} \|\mathbf{u} - \mathbf{x}_i\|_2 \leq (n-1)d \leq nd$.

Based on these two observations, there always exists a point $\mathbf{u} \in L$, for which $\mathcal{J}_2^1(\mathbf{u}) < \mathcal{J}_2^1(\mathbf{v})$ for all $\mathbf{v} \in M$. Hence, we have shown that if there exist a solution for (38) it is found in the compact set L. Since any vector norm is continuous, we know according to Weierstrass' theorem [33, p.654] that there exists a minimizing solution to problem (38) in the compact set L. □

Adapting the ideas of [281] the following basic result is obtained, which proves the uniqueness of the spatial median. The proof is further extended to encompass also missing data cases by Valkonen [381].

Theorem 4.6.2. *If the sample is not collinear, then the spatial median is unique.*

Proof. Based on the previous Theorem 4.6.1 it is known that there exists a solution $\mathbf{u}^*$ for problem (38), which belongs to the compact set L. Let us denote $\alpha := \mathcal{J}_2^1(\mathbf{u}^*)$, i.e. the minimizing value of the cost functional. Concerning uniqueness, assume that there exist two solutions $\mathbf{u}_1$ and $\mathbf{u}_2$ for (38). Then, from the triangle inequality it follows that for all $1 \leq i \leq n$ we have

$$\|\frac{\mathbf{u}_1 + \mathbf{u}_2}{2} - \mathbf{x}_i\|_2 = \frac{1}{2}\|(\mathbf{u}_1 - \mathbf{x}_i) + (\mathbf{u}_2 - \mathbf{x}_i)\|_2 \leq \frac{1}{2}(\|\mathbf{u}_1 - \mathbf{x}_i\|_2 + \|\mathbf{u}_2 - \mathbf{x}_i\|_2).$$

Hence,

$$\begin{aligned} \mathcal{J}_2^1\left(\frac{\mathbf{u}_1 + \mathbf{u}_2}{2}\right) &= \sum_{i=1}^{n} \left\| \frac{\mathbf{u}_1 + \mathbf{u}_2}{2} - \mathbf{x}_i \right\|_2 \\ &\leq \frac{1}{2}\sum_{i=1}^{n} \|\mathbf{u}_1 - \mathbf{x}_i\|_2 + \frac{1}{2}\sum_{i=1}^{n} \|\mathbf{u}_2 - \mathbf{x}_i\|_2 \\ &= \frac{\alpha}{2} + \frac{\alpha}{2} = \alpha, \end{aligned} \tag{40}$$

which shows that also $\frac{\mathbf{u}_1+\mathbf{u}_2}{2}$ is a solution of (38). Moreover, if the sample is not collinear, i.e. not collapsed on the line going through the points $\mathbf{u}_1$ and $\mathbf{u}_2$, there exists at least one point $\mathbf{x}_k$ in the sample such that

$$\begin{aligned} (\mathbf{u}_1 - \mathbf{x}_k, \mathbf{u}_2 - \mathbf{x}_k) &= \|\mathbf{u}_1 - \mathbf{x}_k\|_2 \, \|\mathbf{u}_2 - \mathbf{x}_k\|_2 \cos(\mathbf{u}_1 - \mathbf{x}_k, \mathbf{u}_2 - \mathbf{x}_k) \\ &< \|\mathbf{u}_1 - \mathbf{x}_k\|_2 \, \|\mathbf{u}_2 - \mathbf{x}_k\|_2. \end{aligned} \tag{41}$$

Let us denote $\mathbf{v}_1 = \mathbf{u}_1 - \mathbf{x}_k$ and $\mathbf{v}_2 = \mathbf{u}_2 - \mathbf{x}_k$. From (41) we have

$$\begin{aligned} (\mathbf{v}_1, \mathbf{v}_2) &< \|\mathbf{v}_1\|_2 \|\mathbf{v}_2\|_2 \\ \Leftrightarrow \|\mathbf{v}_1\|_2^2 + \|\mathbf{v}_2\|_2^2 + 2(\mathbf{v}_1, \mathbf{v}_2) &< \|\mathbf{v}_1\|_2^2 + \|\mathbf{v}_2\|_2^2 + 2\|\mathbf{v}_1\|_2 \|\mathbf{v}_2\|_2 \\ \Leftrightarrow \|\mathbf{v}_1 + \mathbf{v}_2\|_2^2 &< (\|\mathbf{v}_1\|_2 + \|\mathbf{v}_2\|_2)^2 \\ \Leftrightarrow \|\mathbf{v}_1 + \mathbf{v}_2\|_2 &< \|\mathbf{v}_1\|_2 + \|\mathbf{v}_2\|_2. \end{aligned}$$

This readily implies that

$$\|(\mathbf{u}_1 - \mathbf{x}_k) + (\mathbf{u}_2 - \mathbf{x}_k)\|_2 < \|\mathbf{u}_1 - \mathbf{x}_k\|_2 + \|\mathbf{u}_2 - \mathbf{x}_k\|_2,$$

which similarly to (40) yields

$$\mathcal{J}_2^1\left(\frac{\mathbf{u}_1 + \mathbf{u}_2}{2}\right) < \alpha.$$

This is a contradiction with the fact that $\mathbf{u}_1$ and $\mathbf{u}_2$ are both solutions of (38). □

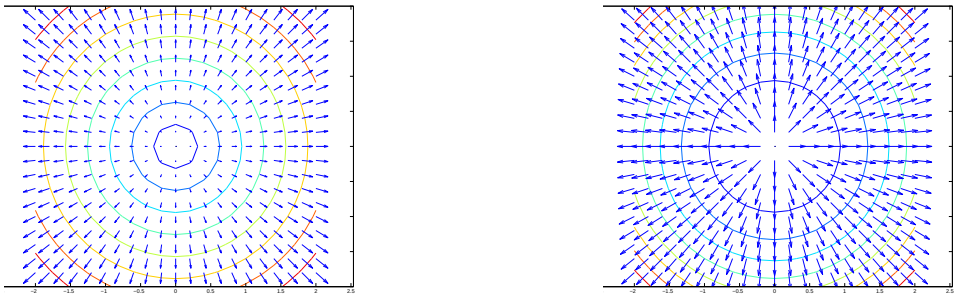


FIGURE 14 Level curves and gradient fields of the squared l_2-norm (left) and l_2-norm (right).

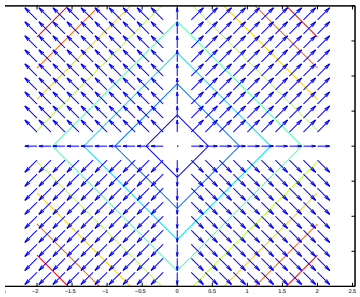


FIGURE 15 Level curves and gradient fields of l_1-norm.

4.6.3 Comparison of statistical and computational properties

Both coordinate-wise and spatial median possess robust nature with a 50% breakdown point that, moreover, is independent of the number of dimensions. The coordinate-wise generalization of the marginal median into p-variate data extends a number of univariate properties to the multivariate case. The spatial median is based on the multivariate generalization of the univariate sign-function $\text{sign}(u)$ with an angular aspect. The breakdown bound of the estimators does not depend on the number of dimensions and equals that of the univariate median [261]. Since there are significant differences between these l_q-norm-based multivariate generalizations of the median, a closer consideration is in order.

In Figures 14 and 15 the bivariate level curves and gradient fields of three l_q-norms are depicted. The level curves illustrate the shape of the corresponding cost function behind the particular M-estimator. The gradient fields illustrate the shape of the bivariate influence functions. Hence, the different measures of robustness can be recovered from the figures.

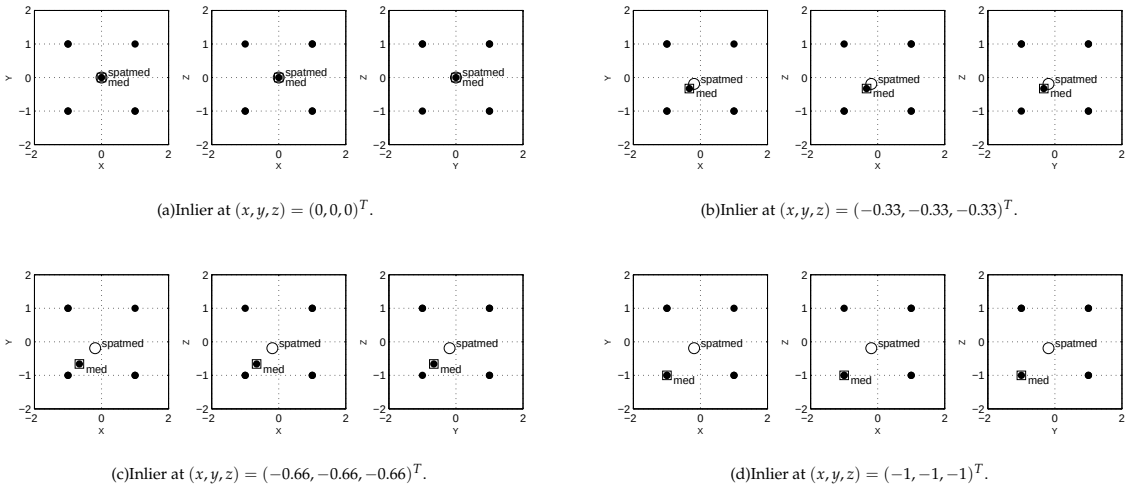
As a comparison, the classical squared l_2-norm is first inspected. Use of this norm leads to the non-robust sample mean estimate, in which case the length of the gradient vectors increases when moving away from the origin. In fact, the length of the gradient vectors grows infinitely, which indicates that the influence function is unbounded (cf. Figures 12-13). This means that the outlying points

are more heavily weighted at equilibrium $\nabla\|\mathbf{x}\|_2^2 = 0$. This readily explains the sensitivity of the squared l_2-estimators towards outliers. Hence, the sample mean is not a robust estimate.

The bounded influence function is obtained by dividing the gradient by its length, which leads to the spatial sign function (cf. (39)) [290]. This is a multivariate generalization of the univariate sign function. Use of the spatial sign function gives equal weights for each observation by ignoring the distance from the data to the equilibrium and, thereby, the corresponding estimation function $\mathcal{J}_2^1$ depends only on the direction of the data (see Figure 14 (right)). Figure 15 clearly illustrates that also the coordinate-wise median gives equal weights for all data. Hence, the l_1-norm is also insensitive to the distance.

Although the spatial median is not affine equivariant, it is still orthogonal equivariant (matrix A in Eq. (4.3.8) must be orthogonal), since the Euclidean norm is invariant under orthogonal transformations. Hence, the Euclidean distances and thereby the spatial median remain invariant after any rotation of the data. From the lack of affine equivariance property, it follows that the scaling of variables may not cause the corresponding effect to the estimate.

The coordinate-wise median is a translation and scale equivariant estimator, because any shifts or scale changes can not alter the marginal orders of points. However, it is not orthogonal equivariant and therefore neither affine equivariant. This makes its use difficult in multivariate problems, provided that data is not discrete. Moreover, the coordinate-wise median does not necessarily lie in the convex hull of the data, provided that the data set lies in $\mathbb{R}^p$ with $p \geq 3$ [338]. For instance, consider a set of unit vectors $[\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n]^T$ of $\mathbb{R}^p$ for $n = p \geq 3$. The coordinate-wise median on such data is $[0, 0, \ldots, 0]^T$, which is not in the convex hull of the unit vector data [338, p.250]. Therefore, it is not a representative vector for the geometric location of such data. On the contrary, the spatial median is always inside the convex hull of a sample.



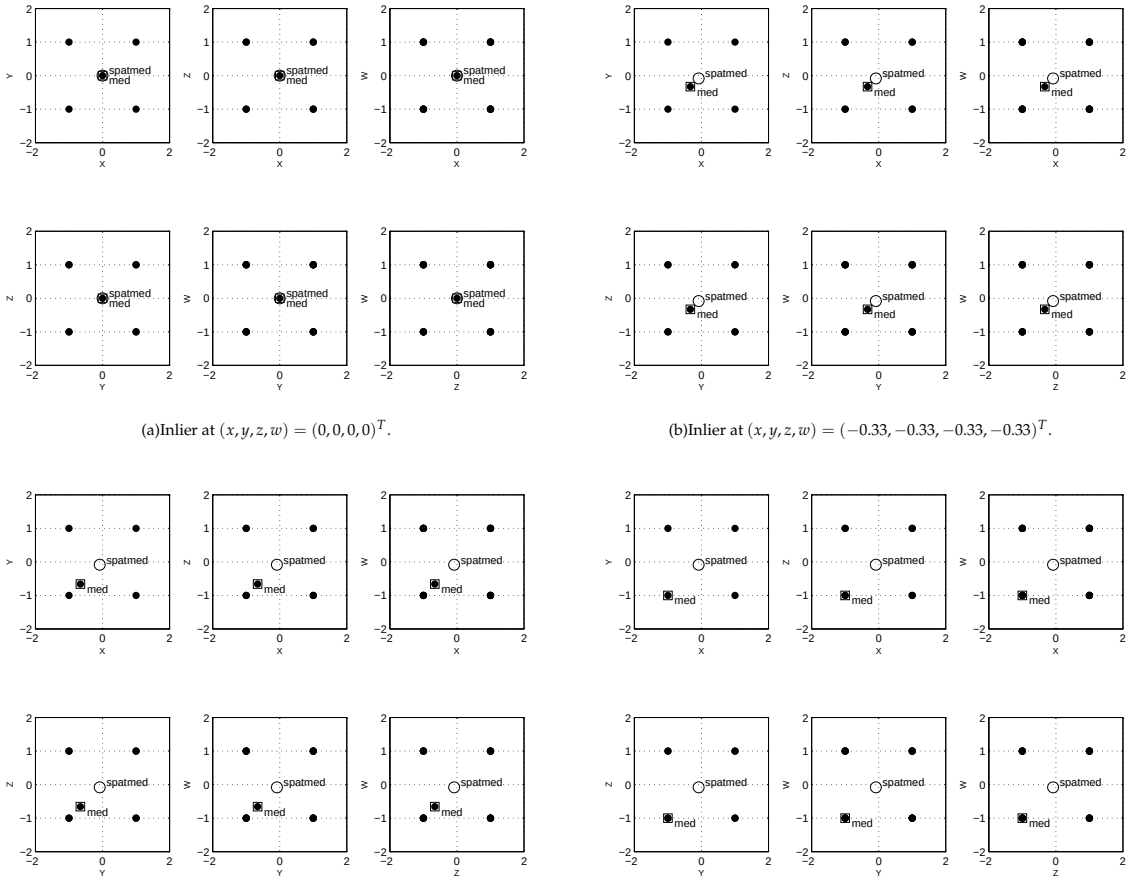
(a)Inlier at $(x, y, z) = (0, 0, 0)^T$.

(b)Inlier at $(x, y, z) = (-0.33, -0.33, -0.33)^T$.

(c)Inlier at $(x, y, z) = (-0.66, -0.66, -0.66)^T$.

(d)Inlier at $(x, y, z) = (-1, -1, -1)^T$.

FIGURE 16 3D comparison of the local-shift sensitivity for the spatial median and the coordinate-wise median estimators.



(a)Inlier at $(x, y, z, w) = (0, 0, 0, 0)^T$.

(b)Inlier at $(x, y, z, w) = (-0.33, -0.33, -0.33, -0.33)^T$.

(c)Inlier at $(x, y, z, w) = (-0.66, -0.66, -0.66, -0.66)^T$.

(d)Inlier at $(x, y, z, w) = (-1, -1, -1, -1)^T$.

FIGURE 17 4D comparison of the local-shift sensitivity for the spatial median and the coordinate-wise median estimators.

The infinite local-shift sensitivity at the median of a distribution makes the coordinate-wise median highly sensitive to one (or two if n is even) middle observations. This feature is illustrated and compared with the spatial median in Figures 16-17. The coordinate-wise median is represented by '□'-mark and the spatial median by '○'-mark. In Figure 16, three-dimensional data ($n = 9$) with eight points in the corners and one at different locations inside the "cube" is represented by two dimensional plots from three perspectives (a kind of scatter plot representation about pairwise relationships between variables [170, p.70]). As the inlying point is moved from the center of the cube towards the corner point in three steps, one can see that the coordinate-wise median is completely defined by this freely moving individual point as far as it lies inside the cube. The problem, for example from the multivariate data clustering point of view, is that on the edge of the cube the "inlier" point is absolutely not the most representative point for the geometric location of the data. In Figure 17 the same illustration is given on four-dimensional hypercube-shape data that consists of seventeen data points. The interesting finding is that while the behavior of the coordinate-wise median is exactly the same as explained above, the spatial median stays at the place even

better. This partly shows that the spatial median benefits from increment in the dimensions.

It is known that spatial median coincides with the coordinate-wise median in the univariate case. From this it follows that it has a unique solution only if the data are not collinear, that is the data are not concentrated on a straight line. In the bivariate case, Brown [54] shows that the spatial median is asymptotically normal on multivariate symmetric spherical data. In the univariate case the normal efficiency naturally coincides with the coordinate-wise median being $2/\pi = 0.637$, but the experiments show that the relative efficiency with respect to the sample mean improves further due to increasing dimensions. Hence, these results and examples indicate that the spatial median is an inherently multivariate estimator.

From the computational point of view, the spatial median estimator suffers from two obvious drawbacks. Lack of the closed-form solution and non-smoothness of the problem have hindered the development of fast and reliable algorithms. The former means that only numerical approximations to the solution are possible and explicit solutions do not exist for the problem [17]. From the DM point of view the algorithms must also scale to large high dimensional data sets. The latter requires reliable algorithmic solutions to the solvers. This problem actually refers to the *inliers* that are data points very close to the solution [55]. In general the ill-defined points of the subgradient function in (39) necessitate the use of approximations in the computation. Incautious approximations in the algorithms may lead to biased estimates and thereby, to lowered consistency, efficiency, and robustness of the estimator when compared to precise theoretical values. Hence, the basic requirements of robust algorithms are at risk. Reasonable efforts have been devoted to the development of fast and correctly converging algorithms for the spatial median problem. Many of these are also referred in this thesis (for a review, see, e.g., [79]). On the contrary to the usual point of view, large DM type data sets and the consequent "curse of dimensionality" turn out to be an advantage in this case, since the increased data sparsity decreases the probability of inliers [55].

4.7 Conclusions

The spatial median seems to have more desirable properties than the coordinate-wise median from the DM point of view. Because DM is focused on heterogenous large-scale data sets, the versatile multivariate properties of the spatial median seem to overcome the coordinate-wise variant. This discussion also reflects consequences of the fact that the coordinate-wise median is actually based on the marginal order-statistics that ignore the angular information about data. On the other hand, such properties make the coordinate-wise median an inherently promising estimator for purely discrete data sets.

The challenge concerning the spatial median is the development of fast and reliable algorithms. One should also be careful with affine transformations that

change the variable scales. Due to the many desirable multivariate properties, the spatial median is considered a promising location estimator for large multidimensional data mining problems. Besides the ones presented here, several other multivariate medians are discussed, for example, in [352, 424].

5 FIRST TESTS ON ROBUST CLUSTERING

In this chapter, robust methods for clustering erroneous and incomplete data sets (without imputation) are considered. For this purpose, the usual K-means algorithm [265] is generalized by using robust location estimates and a special projection technique. Numerical comparison of the resulting methods against the standard K-means algorithm on synthetic data is presented and analyzed. This chapter originates from the results presented in [208].

5.1 Motivation

Very often, data clustering is considered as a core method of DM and KM, and the number of different clustering methods is huge. Based on the previous chapters, we know that data clustering is a challenging task and includes many changeable and adjustable elements, such as the basic algorithmic approach (hierarchical, partitioning, density-based, model-based, grid-based, fuzzy, etc.), the initialization of an algorithm, the distance measure, and the cluster representation technique. These all are dependable on the nature of the particular context where data clustering is intended to be applied. Moreover, as it is described in the previous chapters, the variety of applications is remarkable.

As numerous efficient systems for data gathering have already been developed, there is an obvious need for clustering techniques that are tolerable and reliable on missing and erroneous data (unfortunately most of real-life data is of this form [167, 226]) and scalable to large data sets. Such techniques are of central importance in developing automated "black-box" DM tools that expect a minimal number of parameters and settings from the users. Unfortunately, most of the today's data analysis and mining tools are still based on the tuning and selection of complex parameters before the actual algorithm can be started. An ordinary DM tool customer is usually a specialist on some application domain and not therefore provided with comprehensive skills in computing or statistics. These skills are, however, needed in the parameter adjustment.

Robust techniques are by construction more suitable for erroneous and in-

complete data sets. Therefore, better quality of the results compared to the traditional clustering methods is expected. However, as usually, nothing is for free, since the price to pay for the robustness is usually expressed in increased costs in computation.

The well-known K-means algorithm (see, e.g., [220]) works as a reference method for this study. It is a prototype-based partitioning clustering method, whose popularity is based on the simplicity of the algorithm. While being computationally efficient, K-means is unfortunately also very sensitive to all kind of defects and initial conditions. As described in Chapter 3, many variants of the original MacQueen's type of K-means algorithm have been proposed.

Based on the K-means principles, two robust algorithms will be introduced by replacing the sample mean with a robust multivariate estimator. For handling missing values, the available data strategy is applied in all computation (see Section 3.2.8). Thus, all available data will be utilized. Moreover, no additional parameters are provided. Notice that similar approaches are also presented by Estivill-Castro and Yang [104] and Jörnsten et al. [207], but with different algorithmic details and without taking into account the possibility of missing data values.

5.2 Previous work on robust clustering

As classical statistics, also many clustering methods, such as the K-Means algorithm, are sensitive to erroneous and missing values. Even a small amount of errors and missing values may completely distort the clustering result, and then the "correct" underlying cluster structure of the data set remains uncovered. When compared with estimation of a location or scatter parameter, the problem of dirty data is even worse in data clustering. By considering the influence functions, Garcia et al. [11] show that robustness of a prototype estimate does not necessarily extend to a clustering algorithm. Figures 33 and 34 show that robustness of a partition-based clustering method depends on the initialization, because the global minimum of the cost function is not necessarily the best solution for finding groups' point of view. This means that the initialization must also be robust. In order to increase the robustness of clustering algorithms, Garcia et al. [11] propose a trimmed variant of the classical K-means method.

Lack of statistical robustness is not the only trap for clustering methods. Therefore, one should make a difference between statistical and computational properties. Although the desired statistical properties are obtained, one should also concentrate on reliable computation. If computational and algorithmic details of all elements are not thoroughly considered, hidden problems may remain in the clustering algorithms (e.g., computational issues when dealing with non-smoothness of the spatial median, Section 4.6.2). The computation of EM-algorithm for Gaussian mixture models is a good example. It is based on precise parametric models that exploit a lot of information about location and variabil-

ity of the cluster structure. However, the computation becomes unstable when the cluster-wise variance of a cluster is very close to zero for one or more variables. Such situation may appear especially with categorical variables or missing data. Hence, computational tricks are needed with this statistically fundamental mixture model method (see [308]). The hierarchical single linkage algorithm (e.g., [204]) fails when a data point triplet lies, due to missing values, in different sub-spaces of $\mathbb{R}^p$. In such cases, the order of the mutual distances is undefined without special treatments or distance measures. Overall, the algorithmic details are as important issue as the statistical properties in data clustering methods.

The basic idea of the K-Means method is to partition a given data set into K non-overlapping clusters. The sample mean over the assigned data points is assumed to be the most representative point for each cluster. The two basic steps of the K-means iteration, namely the assignment of points and computation of prototypes, are easy to generalize. Hence, new methods are often developed by modifying these steps. As mentioned earlier, robust variants for the iterative relocation clustering algorithms are obtained by replacing the sample mean with the spatial median [208, 104, 207].

One can also replace the sample mean by the coordinate-wise median [9, p.166]. The coordinate-wise median is as robust as the spatial median, but fits better discrete than continuous data, because it is based on the l_1-norm ('city-block'-distance). Bradley et al. [46, 44] propose a formulation for the problem of the K-coordinate-wise medians as a bilinear programming problem. As a coordinate-wise order-statistic, K-(coordinate-wise) medians is proper for discrete data types, such as questionnaires. The spatial median is more appropriate for continuous multivariate data sets in $\mathbb{R}^p$ with $p \geq 2$. The coordinate-wise median lacks some multivariate properties of the spatial median. A robust location estimator can also be derived from the sample mean by trimming. Hence, one can obtain increased robustness by trimming a certain fraction of cluster-wise data at each iteration, which leads to the trimmed variant of the K-means method [11, 83]. A robust fuzzy c-means method that is based on the same idea is proposed by Butkiewicz [58].

Perhaps the best-known robust variants of the partitioning-based algorithms are so-called *medoid* algorithms: K-medoids [175], PAM [220], CLARA [220], and CLARANS [303]. These are clustering algorithms, where the prototypes are constrained so that they are chosen from the data points (that is $\{\mathbf{m}_k\}_{k=1}^K \subset \{\mathbf{x}_k\}_{i=1}^n$). As the K-medoids approach is more robust against outliers and noise than K-means, it is also computationally more expensive. For example, a fast variant (with sub-quadratic time complexity) of the medoid-based algorithms for Web mining applications is proposed by Estivill-Castro and Yang [103]. Medoid algorithms are invariant to translations and orthogonal transformations of data, but not invariant to affine transformations that lead to changes in the inter-object distances [220, p.119]. The K-medoids algorithm results always in K non-empty clusters. In order to reduce the computational requirements, enhanced variants for the K-medoids algorithms, such as CLASA [71], which is based on the simulated annealing approach, can be considered.

There are also some other approaches related to robust clustering. For example, based on the concepts of mutual information and variance analysis, Fred and Jain [128] introduce an information-theoretic ensemble approach for robust clustering. The principle is to construct several partitions for a given data, analyze the consistency and variability of the obtained partitions and combine them by the so-called evidence accumulation technique. Another robust method that is based on a mutual agreement between several clusterings is the clustering aggregation principle introduced by Gionis et al. [143]. A robust algorithm for spatial clustering is proposed by [102]. One of the most recent efforts on the robust clustering is presented by Gallegos et al. [136]. Robust clustering algorithms have also been integrated into statistical software packages, such as S-PLUS [364].

5.3 Generalization of K-means method

The K-means clustering problem (12) is generalized next for any l_q-norm and missing data. Hence, the general algorithm does not define the type of the distance and prototype estimation functions. A convergence analysis is performed for squared and non-squared l_2-formulations. A similar definition under the name "total within-group distance problem" is presented by Estivill-Castro and Houle [100], but without missing data strategy. They also present a fast and robust clustering algorithm for spatio-temporal data mining problems that avoids quadratic complexity.

Before going to the formulation of the clustering problems, let us say few words about the chosen missing data strategy. In order to deal with incomplete data sets, a strategy for the missing data treatment must be embedded into the formulae. Since it makes no sense from the DM point of view to be involved in making hypotheses on the distributions of unknown cluster-wise data, the chosen missing strategy is to employ only available data values in the calculation of distances and location estimators (see details in Section 3.2.8 or [258, 106]). From this it follows that all computations are restricted to existing fields of the original data. Therefore, the subsequent optimization problems will be generalized for missing data cases by using the projector technique given by (9).

5.3.1 Partitioning-based clustering problems based on l_q-norm and missing data treatment

At first, the generalized K-estimates clustering problem using the projector technique (9) for missing data is defined as

$$\min_{\mathbf{c}\in\mathbb{N}^n,\mathbf{m}_k\in\mathbb{R}^p} \mathcal{J}(\mathbf{c},\{\mathbf{m}_k\}_{k=1}^K) = \sum_{i=1}^{n} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}_{(\mathbf{c})_i})\|_q^{\alpha} \tag{42}$$

$$\text{subject to } (\mathbf{c})_i \in \{1,\ldots,K\} \text{ for all } i = 1,\ldots,n.$$

Code vector $\mathbf{c}$ represents the cluster assignments for each data point $\mathbf{x}_i$. Thus, $\mathbf{m}_{(\mathbf{c})_i}$ is the prototype of the cluster, where data point $\mathbf{x}_i$ is assigned to. The choice of $q = 2$ and $\alpha = 2$ leads to the known K-means problem with available case strategy for missing data. Robust variants are obtained by using $\alpha = 1$. By choosing $q = 1$ and $\alpha = 1$, one obtains the *K-medians* clustering problem (to be exact the K-coordinatewise-medians) and by $q = 2$ and $\alpha = 1$ the *K-spatialmedians* clustering problem is obtained.

5.3.2 General K-estimates algorithm with missing data treatment

The same iterative reassignment and batch-update principle as in Algorithm 3.3.4 can be used to solve any of the aforementioned clustering problems. The generalized algorithm is defined next (notations are similar to the ones in Chapter 3).

Algorithm 5.3.1. General K-estimates algorithm

Required input parameters: $\mathbf{X}$,K, and maxit.

Optional input parameters: Initial prototypes $\{\mathbf{m}_k^0\}_{k=1}^K$ or initial partition in code vector $\mathbf{c}^0$.

Output parameters: $\{\mathbf{m}_k\}_{k=1}^K$ and/or $\mathbf{c}$.

Step 1. (*Initialization*) If initial prototypes $\{\mathbf{m}_k^0\}_{k=1}^K$ are given then go to Step 2. Else if initial partition $\mathbf{c}^0$ is given then go to Step 3. If neither prototypes nor partition is given as input then initialize centers $\{\mathbf{m}_k^0\}_{k=1}^K$, assign each data point $\{\mathbf{x}_i\}_{i=1}^n$ to its closest center, and go to Step 3. Set $t = 0$.

Step 2. (*Reassignment*) Assign each data point $\mathbf{x}_i$ ($i = 1, \ldots, n$) to the closest cluster $\mathcal{C}_k$ ($k \in \{1, \ldots, K\}$), which is given by

$$(\mathbf{c})_i \leftarrow \underset{k \in \{1,\ldots,K\}}{\arg\min} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}_k)\|_q^\alpha.$$

If the minimization results in a tie-break situation with the existing assignment for $\mathbf{x}_i$, its reassignment will be omitted. In other tie-break cases the random selection between the tied cluster centers can be used.

Step 3. (*Recomputation*) If the cluster reassignments were changed in Step 2. and $t <$ maxit, then recompute the prototypes of all modified clusters by

$$\mathbf{m}_k \leftarrow \underset{\mathbf{m}_k}{\arg\min} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}_{(\mathbf{c})_i})\|_q^\alpha$$

and repeat from Step 2. Otherwise, stop.

By varying q and α, different K-estimates algorithms are obtained from Algorithm 5.3.1. The projector technique generalizes the algorithms to missing data cases, which means that preprocessing methods, such as imputation of missing data, are not needed.

5.3.3 Convergence analysis

The convergence proof for the K-means-type of algorithms is given by Selim et al. [344]. Here, by using a new formulation, the proof is extended and generalized for the K-spatialmedians case with missing data treatment. In general, the iterative relocation algorithm does not attain the global minimum over the assignments. Since the algorithm is not based on continuous optimization, but instead on the discrete assignments in Step 2., the attained minimum is not necessarily even a real local minimum of the cost function. However, the obtained local minimum is the smallest permissible point of the particular convex part of the discrete cost function. A similar, but more informal, argumentation about the convergence is given by Verbeek [386]. In the following, the convergence proof is given for the l_2-norm-based algorithms (i.e. $q = 2$ and $\alpha \in \{1, 2\}$), since both cases can be presented in parallel.

Theorem 5.3.1. *From any initial cluster centers, Algorithm 5.3.1 converges in a finite number of steps for choices $q = 2$ and $\alpha \in \{1, 2\}$.*

Proof. Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a random sample in $\mathbb{R}^p$. Let $1 \leq K \leq n$ be the number of clusters. P^t denotes the partition of $\mathbf{X}$ into K non-empty clusters at iteration t. Hence, for P^t it holds that $\mathbf{X} = \cup_{k=1}^{K} \mathcal{C}_k^t$ and $\mathcal{C}_k^t \neq \emptyset$ for all $k = 1, \ldots, K$. The set of vectors $\{\mathbf{m}_k^t\}_{k=1}^K$ represents the prototypes of $\{\mathcal{C}_k^t\}_{k=1}^K$ at iteration t. Function $s(P, i) \in \{1, \ldots, K\}$ returns membership of $\mathbf{x}_i$ with respect to a partition P.

As we know that the number of distinct partitions is finite, it is sufficient to show that by choosing $q = 2$ and $\alpha \in \{1, 2\}$ Algorithm 5.3.1 decreases the cost function values of problem (42) on each iteration. Hence, the optimal partition is attained when the location of each cluster center is optimized (ie., in the subgradient sense $\mathbf{0} \in \nabla_{\mathbf{m}_k^t} \mathcal{J}(P^t, \{\mathbf{m}_k^t\}_{k=1}^K)$) and additional reassignments of the current partition can not decrease the value of the cost function.

According to the steps of Algorithm 5.3.1, the proof will be carried out for $q = 2$ in two phases.

Phase 1.) The cost of the current partition P^t and cluster-wise prototypes $\{\mathbf{m}_k^t\}_{k=1}^K$ is

$$\mathcal{J}(P^t, \{\mathbf{m}_k^t\}_{k=1}^K) = \sum_{i=1}^{n} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}_{s(P^t,i)}^t)\|_2^\alpha.$$

$\{\mathbf{m}_k^t\}_{k=1}^K$ is kept fixed until further notice. Assign each data point $\mathbf{x}_i$ to a new cluster $\mathcal{C}_k$ if

$$\min_{k \in \{1,\ldots,K\}} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}_k^t)\|_2^\alpha < \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}_{s(P^t,i)}^t)\|_2^\alpha \quad \text{for} \quad k \neq s(P^t, i). \tag{43}$$

The inequality is justified as it is defined in the subspace of $\mathbf{x}_i$ and prototype vector $\mathbf{m}_k$ is always complete, that is, both sides are projected to the same components.

Let us denote the reconstructed partition by $\hat{P}^t$. Next we divide $\mathbf{X}$ into two sets $\{S_A, S_I\}$. S_A contains the reassigned (active) points $\{\mathbf{x}_i | s(P^t, i) \neq s(\hat{P}^t, i)\}$ and S_I the rest (inactive points). If $|S_A| > 0$, then clearly

$$\mathcal{J}(\hat{P}^t, \{\mathbf{m}_k^t\}_{k=1}^K) \;<\; \mathcal{J}(P^t, \{\mathbf{m}_k^t\}_{k=1}^K),$$

because by (43)

$$\begin{aligned}
&\sum_{\mathbf{x}_i \in S_A} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}^t_{s(\hat{P}^t,i)})\|_2^\alpha + \sum_{\mathbf{x}_i \in S_I} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}^t_{s(\hat{P}^t,i)})\|_2^\alpha \\
&= \sum_{\mathbf{x}_i \in S_A} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}^t_{s(\hat{P}^t,i)})\|_2^\alpha + \sum_{\mathbf{x}_i \in S_I} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}^t_{s(P^t,i)})\|_2^\alpha \\
&< \sum_{\mathbf{x}_i \in S_A} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}^t_{s(P^t,i)})\|_2^\alpha + \sum_{\mathbf{x}_i \in S_I} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m}^t_{s(P^t,i)})\|_2^\alpha.
\end{aligned}$$

If $|S_A| = 0$, then we are done.

Phase 2.) For each $\mathcal{C}_k^t$, new center $\hat{\mathbf{m}}_k^t$ is obtained by minimizing the sum of the error function as

$$\hat{\mathbf{m}}_k^t \leftarrow \arg\min_{\mathbf{m}} \mathcal{J}(\mathbf{m}), \quad \text{for} \quad \mathcal{J}(\mathbf{m}) = \sum_{\mathbf{x}_i \in \mathcal{C}_k^t} \|\mathbf{P}_i(\mathbf{x}_i - \mathbf{m})\|_2^\alpha. \tag{44}$$

This leads to special cases of M-estimation for l_2-norm with the available case strategy for missing data. The minimizing point is the mean or the spatial median of given data vectors.

Since the within-cluster sum of the absolute or squared l_2 distance is minimized for each cluster after reassignments, the total sum can only decrease from the first phase. Hence,

$$\mathcal{J}(\hat{P}^t, \{\hat{\mathbf{m}}_k^t\}_{k=1}^K) \;\leq\; \mathcal{J}(\hat{P}^t, \{\mathbf{m}_k^t\}_{k=1}^K).$$

Assuming that there exists a minimizing solution for the sample mean and spatial median with the available case strategy, the new prototypes $\hat{\mathbf{m}}_k^t$ do not increase the value of the clustering criterion. Now, $P^{t+1} = \hat{P}^t$ and $\{\mathbf{m}_k^{t+1}\}_{k=1}^K = \{\hat{\mathbf{m}}_k^t\}_{k=1}^K$. Because by every reassignment made according to the first phase, the value of $\mathcal{J}(P^t, \{\mathbf{m}_k^t\}_{k=1}^K)$ is decreased and this can not be increased in the second phase, it follows that $\mathcal{J}(P^{t+1}, \{\mathbf{m}_k^{t+1}\}_{k=1}^K) < \mathcal{J}(P^t, \{\mathbf{m}_k^t\}_{k=1}^K)$, whenever one or more points are reassigned during the t^{th} iteration. Since there exists only a finite number of different partitions, this shows the result. □

5.4 Statistical experiments on synthetic data sets

The three variants of the K-estimates clustering algorithm, K-means, K-medians, and K-spatialmedians, are next tested on synthetic bivariate data sets. The data

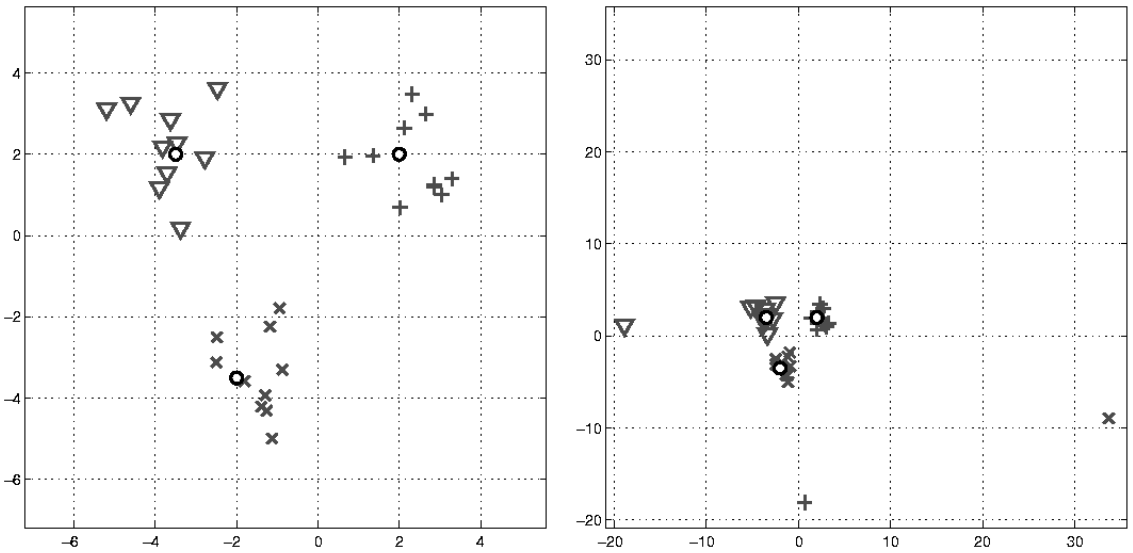


FIGURE 18 Left: A plot of the test data set, in which three clusters of size ten are well-separated. Right: The dirty test data set, in which four data values (appr. 6.67 percent of the whole data) are randomly distorted.

sets with size $n = 30$ and dimension $p = 2$ are random samples from trimodal Gaussian distributions. The clusters are clearly separated (see Figure 18). In order to investigate the sufficiency and consistency of the methods under nearly optimal conditions (well-separated spherical clusters without defects), the first experiments were performed on complete and clean data set. For analyzing the robustness of the different methods to outliers and missing data, four randomly chosen data values were disturbed and, thereby, turned over to outliers (see Figure 18). Moreover, 10%, 30% and 50% of data values, in turn, were removed.

The experiments were realized on MATLAB 6.1. environment. The spatial median was computed using the self-implemented Polak-Ribiere-type conjugate gradient (CG) optimization method where golden section (GS) was used for determining the search step size [28]. Because the CG method utilizes the derivative information of the cost function and the gradient of the spatial median cost function is not well-defined everywhere, the obtained CG solutions were further fine-tuned by the simplex-based Nelder-Mead algorithm [242].

The maximum norm of cluster displacement

$$\max_{k\in\{1,\ldots,K\}} \|\mathbf{m}_k^t - \mathbf{m}_k^{t-1}\|_\infty \leq \epsilon$$

was used as a stopping criterion. The tolerance parameter was $\epsilon = 10^{-3}$. The stopping criteria for CG was chosen to be $\|\mathbf{u}^t - \mathbf{u}^{t-1}\|_\infty \leq 10^{-6}$ and, in GS, $\|\mathbf{u}^t - \mathbf{u}^{t-1}\|_2 \leq 10^{-8}$, where $\mathbf{u}^t$ is the solution after t^{th} iteration.

The results were obtained by running the algorithm for 100 random initial cluster prototypes on each data sets. The efficiency (likelihood of unbiased solutions) of the methods in the statistical sense was analyzed using visual histogram presentations for the error distributions (over 100 test runs).

The error estimates are defined by the sum of distances from the obtained

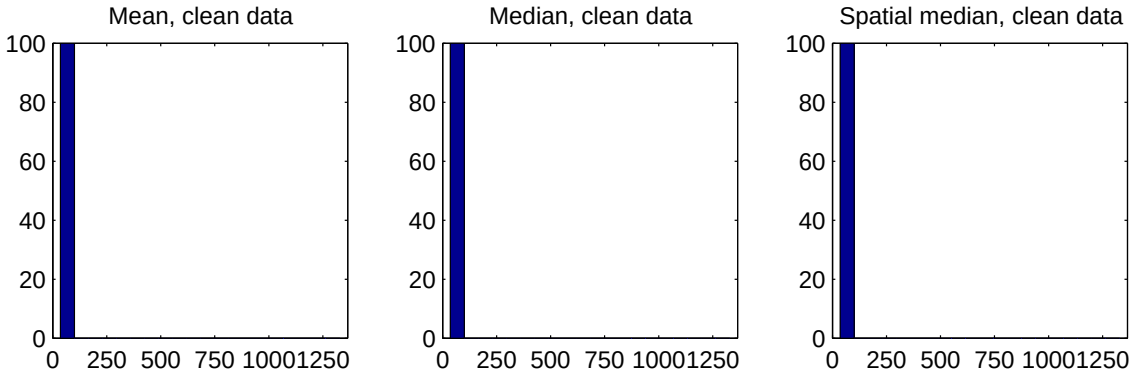


FIGURE 19 Error distributions of the 100 test runs on the complete non-disturbed data set.

cluster centers to the centers of the generating distribution. Let $\boldsymbol{\mu}_k \in \mathbb{R}^p$ $(k = 1, \ldots, K)$ be the true means of the generating distribution modes and $\mathbf{m}_k \in \mathbb{R}^p$ $(k = 1, \ldots, K)$ the ones obtained by a chosen clustering algorithms. Let $\pi(k)$ denote any permutation on $\{1, ..., k\}$. The error can then be determined by finding the permutation $\pi(k)$ that minimizes the distance from the true centers $\boldsymbol{\mu}_k$ to the ones computed by the clustering algorithms:

$$err = \min_{\pi(K)} \sum_{k=1}^{K} \|\boldsymbol{\mu}_k - \mathbf{m}_{\pi(K)}\|_2^2. \tag{45}$$

The permutation is needed, because the generated prototypes can be in any order with respect to each other. The mean estimate $\widehat{err}$ for the error on a particular method is computed as the average error of 100 trials. The median estimate is computed correspondingly.

5.4.1 Results

Let us finally consider the results. As it is shown in Figure 19, no significant differences in the statistical efficiency of the K-means, K-medians, and K-spatialmedians methods was found on the complete non-disturbed data set. All the algorithms seem to produce very good results under such conditions.

Next, the clustering algorithms were tested on a data set that contains some outliers, but no missing data. The results are illustrated in Figure 20. As it can be seen, the statistical efficiency of the algorithms was decreased due to the outliers. Moreover, clear differences can be observed when comparing the results of the K-means algorithm to the ones obtained by the robust algorithms. K-means leads to approximately reasonable results in 50% probability, whereas for K-medians and K-spatialmedians the same number is almost 90% with almost identical performance.

Removing 10% of data seems to increasingly impair the performance of K-means (see Figure 21). Approximately half of the test runs produced significant errors. The statistical efficiency of K-medians and K-spatialmedians also decreased when compared to the complete data cases with outliers, but approx-

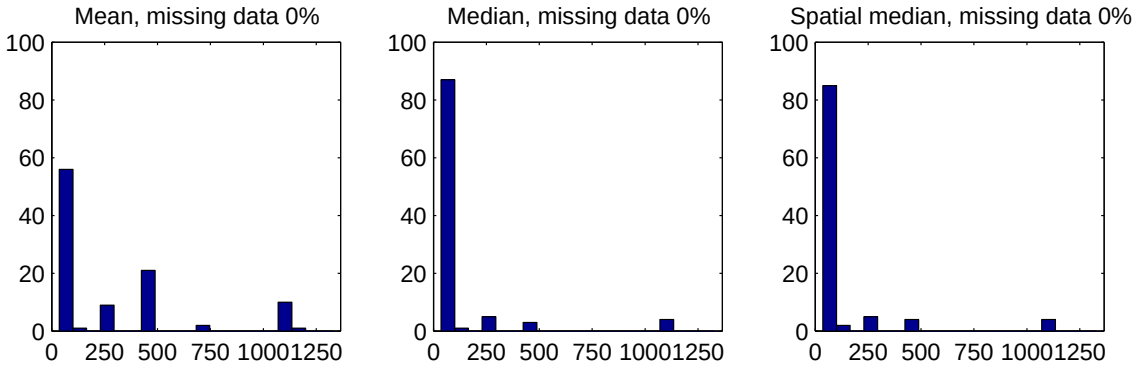


FIGURE 20 Error distributions of the 100 test runs on the complete data set with outliers.

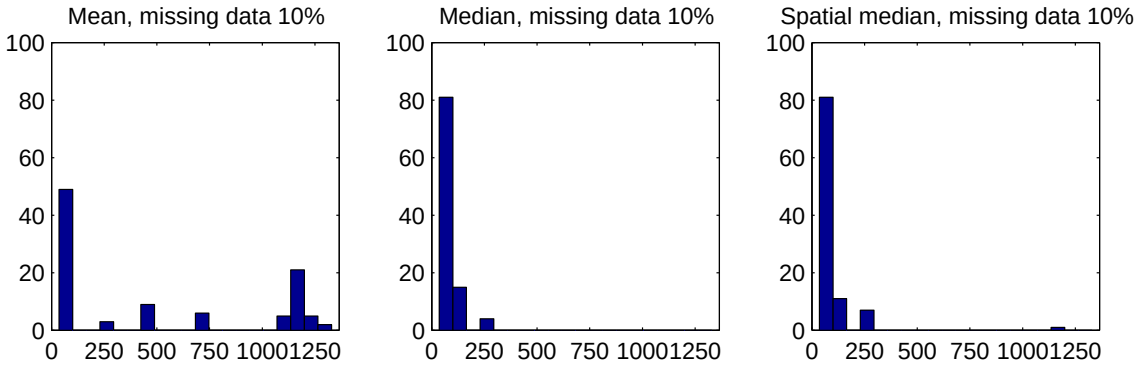


FIGURE 21 Error distributions of the 100 test runs on the incomplete data set (10% of values missing) with outliers.

imately 80% of the results were still of good quality and almost none were poor. No significant differences between the robust algorithms were found.

To break also the robust algorithms, the amount of missing data was increased to 30% of the total. The results are illustrated in Figure 22. Obviously, the statistical efficiency of K-means is not anymore feasible. Nearly all of the results are unsatisfactory. The results obtained by K-medians and K-spatialmedians were again quite identical to each other. The results were even better when compared to the two previous cases, but this is considered as a coincidence. However, it shows that even a third of the data may be lost without a significant influence on the performance of the robust algorithms.

Finally, Figure 23 presents the mean and median estimates for the clustering errors with the different methods. Considering the median estimates of the K-means clustering errors, one can observe that one half of the clustering solutions are of relatively poor quality when more than 10% of data is missing and outliers are present.

The contribution of the outliers to the performance of K-means may be deduced by comparing its error estimates with the robust algorithms. Figure 23 shows that for K-means the error, when no missing data exist in the data, is greater than the error of K-medians and K-spatialmedians when 10%-50% of data

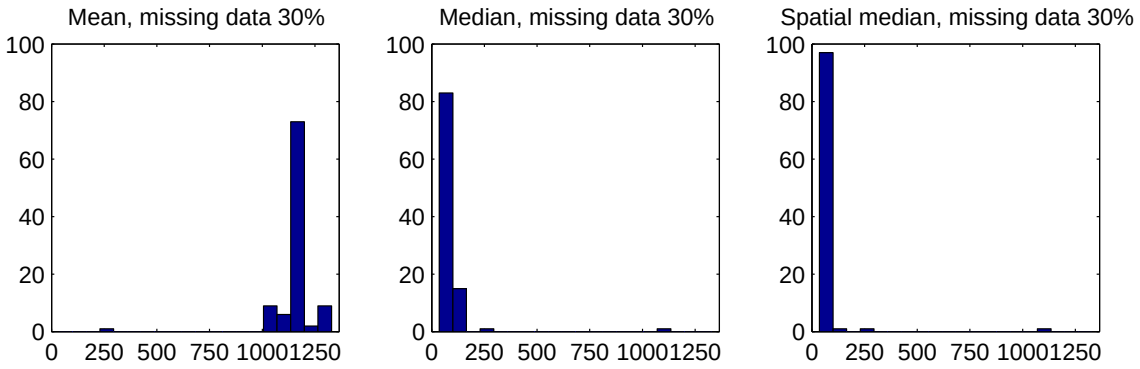


FIGURE 22 Error distributions of 100 test runs on the incomplete data set (30% of values missing) with outliers.

is missing.

Although the median errors of K-medians and K-spatialmedians do not increase much with respect to the amount of missing data, the average quality of the results is impaired slightly.

5.5 Conclusions

The data mining algorithms inherently contain a number of adjustable parameters that often are difficult to understand by an ordinary end user. Therefore, algorithms that are adjusted in advance to the target environment are needed. The ultimate goal of this thesis is to make a contribution to the development of robust data mining algorithms that work well even on noisy and incomplete data sets without a number of user-defined parameters. Although estimation of the correct number of clusters has not yet been considered in this chapter, it is one of the main issues related to data clustering [220].

The main goal of this chapter is to report on preliminary tests for the robust estimators that are used together with a special missing data treatment in iterative prototype-based clustering algorithms. Other issues, such as initialization, computational efficiency, and scalability are considered in other parts of this thesis. However, the obtained results are encouraging, since the robust algorithms performed much better on the erroneous and missing data sets. A large part of the test runs resulted in good quality of solutions for the robust methods, even when 10%-50% of the data were missing and 7% distorted. Hence, it is expected, and for good reasons, by developing efficient and robust initialization methods for clustering algorithms, one is able to cluster complex data sets without prior operations, such as outlier detection or imputation. The performance of K-spatialmedians and K-medians differed slightly from each other. This observation is not considered completely unexpected, because all the test test data sets were sampled from bivariate distributions. The inconsistencies of the coordinate-wise sample median estimator appear in higher dimensional problems. More-

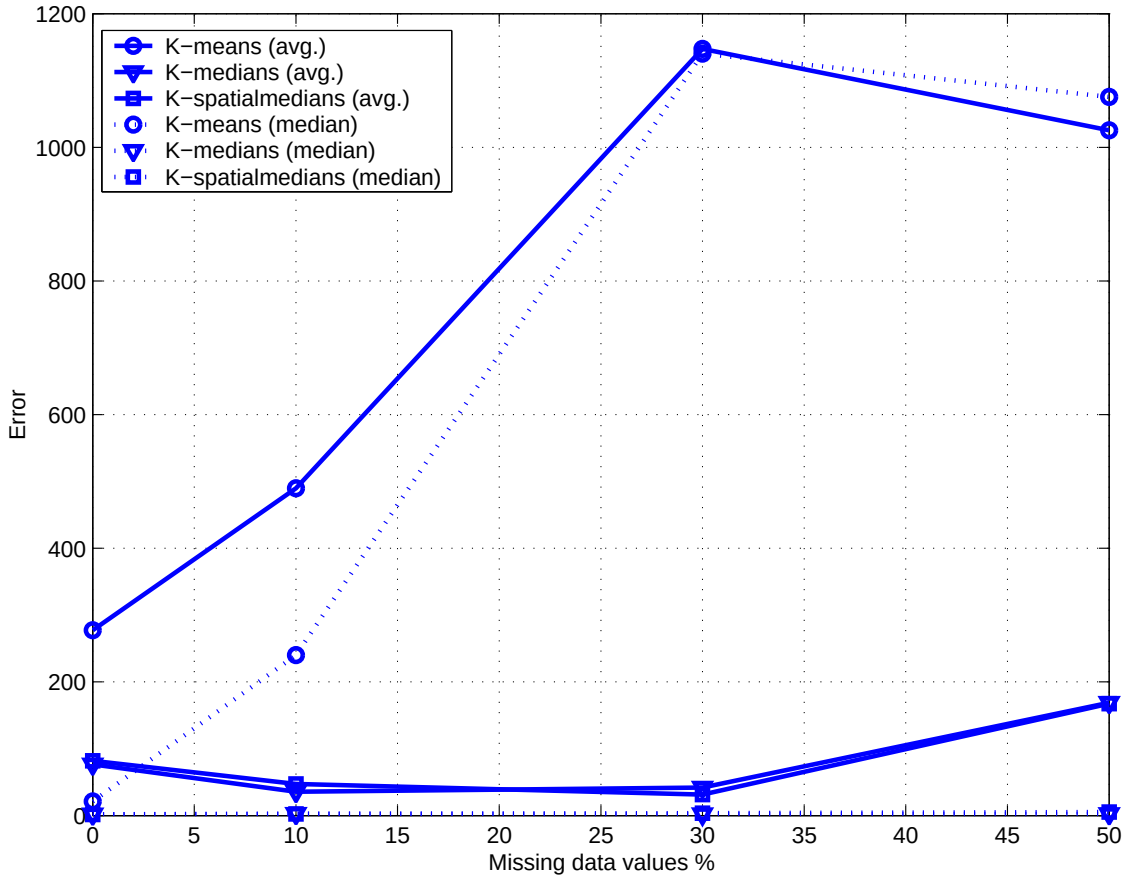


FIGURE 23 The mean and median estimates of the clustering errors for the different methods.

over, as the coordinate-wise median is based on the l_1-norm, it is better applied to the discrete data problems. In these experiments, the algorithms used for solving the problem of spatial median are not scalable to higher dimensions and data size. More efficient methods with missing data treatment are developed and presented in the subsequent parts of this thesis.

6 FAST COMPUTATION OF ROBUST LOCATION ESTIMATES

Even if robustness itself has been a major issue since the sixties, not so many robust tools for data mining or analysis tasks have been developed. This is probably due to the difficulty of the formulations and complexity of the algorithms. While the sample mean is computed in short time, robust estimates often involve trimming and sorting operations, or non-smooth optimization problems that require iterative algorithms for solving. Hence, a serious problem for the robust estimation are the computational details. For instance, incautious approximations in computer implementations may lead to inaccurate solutions for a theoretically robust, consistent, and efficient estimator. Also wrong assumptions about mathematical formulations increase the risk of such errors. This means that variability of the estimates grows, which leads to uncertainty of the results. On the other hand, algorithmic solutions require thorough analytical considerations and testing under various conditions in order to be verified and validated.

Hence, there is a lot of work to do for the statisticians and computer scientists in making robust techniques feasible, for example, in unsteady large-scale problems that are frequently faced in the field of data mining and knowledge discovery. Previous efforts on developing fast algorithms for robust estimation of location in the fields DM and KDD can be found, e.g., in [101]. The paper presents randomized estimators of location for approximating in subquadratic time two robust and orthogonal equivariant estimators: the least median square and least trimmed square [338].

In this chapter, an effort to the development of the robust data mining methods is given. A number of classical optimization methods, special iterative methods, and different formulations for solving the problem of the spatial median are introduced and compared. Numerical experiments are performed in order to evaluate the accuracy of the solutions and the computational requirements. The algorithms and problem formulations include the handling for missing data. Statistical experiments are performed in order to evaluate how the statistical properties are maintained with respect to missing data.

The contributions of this chapter are as follows:

1. A new formulation for the problem of spatial median taking into account missing values.

2. A new algorithm for computing the spatial median and comparison to (many) others.

3. Tests of both statistical and computational properties.

6.1 Iterative methods for solving the problem of spatial median

Some proposals for the use of spatial median in the fields of DM and KDD are given, e.g., in [104]. It is also well-known that there exist no explicit closed-form solutions to the problem of the spatial median and only numerical approximations to the solution are possible [17]. The best-known algorithm for solving the problem of the spatial median is the Weiszfeld algorithm. The algorithm was first proposed by Weiszfeld in the 1930s[1] and has been rediscovered several times since (see, e.g., [241, 262, 78, 279]). It is based on the first-order necessary conditions for a stationary point of the cost function in (38), which provides the following iterative scheme:

$$\mathbf{u}^{t+1} = \begin{cases} \frac{\sum_{i=1}^{n} w_i \mathbf{x}_i / \|\mathbf{u}^t - \mathbf{x}_i\|_2}{\sum_{i=1}^{n} w_i / \|\mathbf{u}^t - \mathbf{x}_i\|_2}, & \text{if} \quad \mathbf{u}^t \notin \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}, \\ \mathbf{u}^t, & \text{if} \quad \mathbf{u}^t = \mathbf{x}_i \quad \text{for some} \quad i = 1, \ldots, n. \end{cases} \tag{46}$$

In this thesis, $w_i = 1$ for all $i = 1, \ldots, n$, since we treat all data points always with equal weights. Therefore, the weights are skipped from the rest of the formulae. $\mathbf{u}^{t+1} = \mathbf{u}^t$ has been defined in order to make the scheme defined and continuous for all $\mathbf{x} \in \mathbb{R}^p$. Kuhn [241] proved in 1973 that the Weiszfeld algorithm (globally) converges to a unique minimizing point $\mathbf{u}^*$, for all but a denumerable number of initial points $\mathbf{u}^0$, assuming that the data is not collinear. Katz [219] derived results that when the minimizing point $\mathbf{u}^*$ is not any of the data points, the local convergence of the Weiszfeld algorithm is always linear. Otherwise, when optimal point $\mathbf{u}^*$ coincides with a data point, then the local convergence can be linear, quadratic or sublinear. He proposed the use of Steffensen's iteration in order to assure a quadratic convergence speed.

In 1989 Chandrasekaran and Tamir questioned in [64] the claim that the non-collinearity is not necessarily a sufficient condition for the Kuhn's convergence theorem. They used counter-examples to demonstrate that the Weiszfeld algorithm may not convergence for continuous sets of starting points when the points are contained in an affine subspace of $\mathbb{R}^p$. In other words, this meant that the non-collinearity had to be replaced by the more stringent assumption that the convex hull of the data points $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^p$ is of full-dimension p. This conjecture was later resolved and formally proved by Brimberg in [47] in

[1] The original paper of the Weiszfeld [396] was not available to the author of the thesis, but a plenty of citations can be found from articles.

1995, until Cánovas et al. [59] discovered mistakes in the proof, which opened the question again in 2002. An attempt to sidestep the problem is proposed in a recent report by Rautenbach et al. [328] in which they introduce a non-continuous modification of the Weiszfeld iteration with a proof of convergence to the optimal solution from any starting point without any further assumptions.

Another modification to the Weiszfeld algorithm was proposed by Vardi et al. [382, 383] in 2000 with the guarantee of monotonical convergence to the spatial median from any starting point in $\mathbb{R}^p$. In order to represent this modified Weiszfeld iteration the following definitions are given

$$\eta(\mathbf{u}) = \begin{cases} 1, & \text{if} \quad \mathbf{u} \in \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}, \\ 0, & \text{otherwise}, \end{cases}$$

$$r(\mathbf{u}) = \Big\| \sum_{\mathbf{x}_i \neq \mathbf{u}} \frac{\mathbf{x}_i - \mathbf{u}}{\|\mathbf{x}_i - \mathbf{u}\|_2} \Big\|_2.$$

The spatial median is attained by the following iterative process

$$\mathbf{u}^{t+1} = \max\Big(0, 1 - \frac{\eta(\mathbf{u}^t)}{r(\mathbf{u}^t)}\Big)\tilde{T}(\mathbf{u}^t) + \min\Big(1, \frac{\eta(\mathbf{u}^t)}{r(\mathbf{u}^t)}\Big)\mathbf{u}^t, \tag{47}$$

where

$$\tilde{T}(\mathbf{u}) = \Big\{ \sum_{\mathbf{x}_i \neq \mathbf{u}} \frac{1}{\|\mathbf{u} - \mathbf{x}_i\|_2} \Big\}^{-1} \sum_{\mathbf{x}_i \neq \mathbf{u}} \frac{\mathbf{x}_i}{\|\mathbf{u} - \mathbf{x}_i\|_2}.$$

The generalization of the Weiszfeld algorithm for more general l_q-distances is presented in [50] with a proof about the global convergence for any q in the closed interval $[1, 2]$, and provided with same assumptions as Kuhn in [241]. In [289, 288, 335] modifications and convergence results of the generalized problem that utilizes an approximating cost function are studied. More results on the location problem of the spatial median and the Weiszfeld algorithm with generalized l_q-norms are given, e.g., in [262, 48, 51]. Note that Cánovas et al. [59] questioned the validity of the convergence results in [48], since they were based on the invalidated results presented in [47]. Uster et al. [380] have studied the convergence of Weiszfeld algorithm when $q > 2$ and introduced a stepsize factor, which makes the iterative procedure converge for such parameter values. Other recent approaches to the problem are, e.g., a Newton-based approach [249].

Acceleration of iterative methods

Since the basic iterative methods, such as the Weiszfeld algorithm, may suffer from the slow convergence speed, several acceleration methods for speeding up the convergence of such algorithms have been investigated. Methods for accelerating convergence of iterative methods are described, for instance, in [56, pp.68-72]. Katz [219] suggested the use of Steffensen's scheme to accelerate the linear convergence speed of the Weiszfeld algorithm to quadratic. Steffensen's iteration is not known to be globally convergent, but it may be used to accelerate the local convergence. However, the improvement with respect to the total CPU time

has been questioned (see, e.g., [387, 49]) due to the increased computation time required by the Steffensen's iteration itself. Another reference on the use of Steffensen's iteration and its speeding effect on the convergence rate with respect to the number of iterations is given in [245]. Drezner [90] and Verkhovsky et al. [387] introduce other variable step size factors for accelerating the Weiszfeld-type iterative procedures and report significant enhancements in the number of iterations. Drezner proposes an acceleration factor that is based on the Aitken's δ^2 process and reduces the running time of the Weiszfeld procedure, at least, by a factor of two. Drezner's acceleration method is reported to be especially efficient for small data sets. About equal reduction in the number of iterations was attained by the FASFL algorithm [387]. For small samples the factor is somewhat smaller than two and for large greater than two. This iterative algorithm is proposed by Verkhovsky and Polyakov, who critize the increased computational complexity (double cost for each iteration) of the Drezner's method. In FASFL, coordinate-wise acceleration factors are used, which means that a converging sequence may form a curved trajectory. Li proposes in [253] an N-Weiszfeld algorithm by including transition from a Weiszfeld step to a Newton step of the system of nonlinear equations. This is somewhat different approach when compared to the previous ones. More general case of accelerating the Fermat-Weber problem with l_q-distances is considered by Brimberg et al. [49]. They present a procedure for determining the acceleration factor as a function of q.

6.2 Reformulation of spatial median problem

In order to solve (38) by the usual gradient-based optimization or iterative numerical methods, approximated and differentiable reformulations can be used. In the following, two smoothed approximating formulations for the problem are proposed.

Modified gradient

As the cost function of problem (38) is potentially problematic only at a finite number of points, well-definiteness can be obtained by disturbing the sub-differential with a small constant (e.g., $\varepsilon = 10^{-8}$), whenever a data point is an inlier, that is, too close to a solution (cf. Section 4.6.3). Hence, a well-defined gradient reads as

$$\nabla \mathcal{J}_2^1(\mathbf{u}) = \sum_{i=1}^{n} \xi_i, \quad \text{with } (\xi_i)_j = \frac{(\mathbf{u} - \mathbf{x}_i)_j}{\max\{\|\mathbf{u} - \mathbf{x}_i\|_2, \varepsilon\}} \quad \text{for all} \quad \mathbf{x}_i. \tag{48}$$

By exploiting a convention that $\frac{0}{0} = 0$, this formula actually equals the treatment where the points that are too close to the solution, are left out from the cost function and its gradient (cf. the extended definition of the gradient by Kuhn [241]). This is reasonable, because the inliers do not significantly contribute to the value of the cost function (for all $\mathbf{x}_i$ ($i \in \{1, \ldots, n\}$) that are close to solution $\mathbf{u}$, it follows

that $\|\mathbf{x}_i - \mathbf{u}\|_2 \approx 0$). Hence, by utilizing this approximated gradient given by (48), for instance, the gradient-based optimization methods, such as the CG algorithm, can be applied to the non-smooth problem (38).

ϵ-approximating problem

In order to avoid problems caused by the non-smoothness of problem (38), small and positive ε can be introduced into the problem similarly to [335, 262, 289, 216]. This leads to the following perturbed, but consequently smooth, ϵ-approximating problem formulation

$$\min_{\mathbf{u}\in\mathbb{R}^p} \mathcal{J}_\varepsilon(\mathbf{u}), \quad \text{for } \mathcal{J}_\varepsilon(\mathbf{u}) = \sum_{i=1}^{n} \sqrt{\|\mathbf{u} - \mathbf{x}_i\|_2^2 + \varepsilon}, \tag{49}$$

where ε is a smoothing constant (e.g., $\varepsilon = 10^{-8}$). According to [289], the approximating cost function converges uniformly to the original one as $\varepsilon \to 0$. As a result, the gradient is well-defined everywhere and reads as

$$\nabla \mathcal{J}_\varepsilon(\mathbf{u}) = \sum_{i=1}^{n} \xi_i, \quad \text{where } (\xi_i)_j = \frac{(\mathbf{u} - \mathbf{x}_i)_j}{\sqrt{\|\mathbf{u} - \mathbf{x}_i\|_2^2 + \varepsilon}}. \tag{50}$$

The smooth perturbed problem can then be solved using gradient-based optimization methods or iterative algorithms. The approximation problem together with the Weiszfeld algorithm has been applied, e.g., to single and multi-facility location problems (see [289, 335]).

6.3 SOR accelerated iterative methods for computation of spatial median on incomplete data

In this section, two variants of the SOR accelerated iterative algorithm for solving (38) are introduced. In order to avoid the problems concerning inconsistent properties of the estimator, an inlier elimination approach and the modified cost function (49) are utilized.

6.3.1 SOR accelerated Weiszfeld algorithm for the perturbed problem formulation with missing data treatment

The basic iteration is based on the first-order necessary conditions for a stationary point of the cost function of the perturbed problem (49):

$$\sum_{i=1}^{n} \frac{\mathbf{u} - \mathbf{x}_i}{\sqrt{\|\mathbf{u} - \mathbf{x}_i\|_2^2 + \varepsilon}} = 0. \tag{51}$$

In order to deal with incomplete data, the perturbed algorithm is generalized by available case strategy for missing data. This is realized by redefining the

perturbed problem by using the projector technique as given in (9). Hence, the perturbed condition in (51) is redefined as:

$$\sum_{i=1}^{n} \frac{\mathbf{P}_i(\mathbf{u} - \mathbf{x}_i)}{\sqrt{\|\mathbf{P}_i(\mathbf{u} - \mathbf{x}_i)\|_2^2 + \varepsilon}} = 0. \tag{52}$$

First, (52) is "linearized" by defining explicit weights using the denominator:

$$\alpha_i^t = \frac{1}{\sqrt{\|\mathbf{P}_i(\mathbf{u}^t - \mathbf{x}_i)\|_2^2 + \varepsilon}}. \tag{53}$$

Assuming that sample $\mathbf{X}$ does not contain empty columns (that is a variable without any values), the candidate solution $\mathbf{v}$ is solved, by combining (52) and (53), from

$$\begin{aligned} &\sum_{i=1}^{n} \alpha_i^t \mathbf{P}_i(\mathbf{v} - \mathbf{x}_i) = 0 \\ \Leftrightarrow &\left(\sum_{i=1}^{n} \alpha_i^t \mathbf{P}_i\right) \mathbf{v} = \sum_{i=1}^{n} \alpha_i^t \mathbf{P}_i \mathbf{x}_i \\ \Leftrightarrow &\mathbf{v} = \left(\sum_{i=1}^{n} \alpha_i^t \mathbf{P}_i\right)^{-1} \sum_{i=1}^{n} \alpha_i^t \mathbf{x}_i. \end{aligned} \tag{54}$$

The obtained solution is then accelerated using the SOR type stepsize factor as follows

$$\mathbf{u}^{t+1} = \mathbf{u}^t + \omega(\mathbf{v} - \mathbf{u}^t), \quad \omega \in [0, 2], \tag{55}$$

where ω is the stepsize factor, $(\mathbf{v} - \mathbf{u}^t)$ is the search direction, and $\mathbf{v}$ is an approximate solution to (38) obtained from (54). The overall algorithm is given as:

Algorithm 6.3.1. SOR

Step 1. Initialize $\mathbf{u}$ and set ω.

Step 2. Solve α_i^ts for $i = 1, \ldots, n$ using (53).

Step 3. Solve the basic iterate $\mathbf{v}$ from (54).

Step 4. Accelerate $\mathbf{u}$ using (55).

Step 5. If the stopping criterion is satisfied then stop, else return to step 2.

Proof of convergence

In order to obtain convergence, the basic iteration (i.e. without the acceleration step (55)) of the perturbed algorithm 6.3.1 must decrease the value of the strictly convex function (49). This is shown by proving the next theorem.

Theorem 6.3.1. *If* $\mathbf{v} \neq \mathbf{u}^t$, *then* $\mathcal{J}_\varepsilon(\mathbf{v}) < \mathcal{J}_\varepsilon(\mathbf{u}^t)$.

Adapting the ideas of Kuhn [241], the above theorem is proven herein.

Proof. $\mathbf{v}$ is the candidate solution for the next iterate as given in (54). Then it is the unique minimum of the strictly convex cost function

$$\tilde{\mathcal{J}}(\mathbf{v}) = \sum_{i=1}^{n} \frac{\|\mathbf{v} - \mathbf{x}_i\|_2^2 + \varepsilon}{\sqrt{\|\mathbf{u}^t - \mathbf{x}_i\|_2^2 + \varepsilon}}, \tag{56}$$

which is obtained from (49). Let us denote $e_i(\mathbf{u}) = \sqrt{\|\mathbf{u} - \mathbf{x}_i\|_2^2 + \varepsilon}$. Since $\mathbf{u}^t \neq \mathbf{v}$,

$$\tilde{\mathcal{J}}(\mathbf{v}) = \sum_{i}^{n} \frac{e_i^2(\mathbf{v})}{e_i(\mathbf{u}^t)} < \tilde{\mathcal{J}}(\mathbf{u}^t) = \sum_{i}^{n} \frac{e_i^2(\mathbf{u}^t)}{e_i(\mathbf{u}^t)} = \sum_{i}^{n} e_i(\mathbf{u}^t) = \mathcal{J}_\varepsilon(\mathbf{u}^t).$$

On the other hand,

$$\begin{aligned} \tilde{\mathcal{J}}(\mathbf{v}) &= \sum_{i}^{n} \frac{\{e_i(\mathbf{u}^t) + [e_i(\mathbf{v}) - e_i(\mathbf{u}^t)]\}^2}{e_i(\mathbf{u}^t)} \\ &= \mathcal{J}_\varepsilon(\mathbf{u}^t) + 2\mathcal{J}_\varepsilon(\mathbf{v}) - 2\mathcal{J}_\varepsilon(\mathbf{u}^t) + \sum_{i=1}^{n} \frac{[e_i(\mathbf{v}) - e_i(\mathbf{u}^t)]^2}{e_i(\mathbf{u}^t)}. \end{aligned}$$

Further,

$$\begin{aligned} \tilde{\mathcal{J}}(\mathbf{v}) = \mathcal{J}_\varepsilon(\mathbf{u}^t) + 2\mathcal{J}_\varepsilon(\mathbf{v}) - 2\mathcal{J}_\varepsilon(\mathbf{u}^t) + \sum_{i=1}^{n} \frac{[e_i(\mathbf{v}) - e_i(\mathbf{u}^t)]^2}{e_i(\mathbf{u}^t)} &< \mathcal{J}_\varepsilon(\mathbf{u}^t) \\ \Leftrightarrow 2\mathcal{J}_\varepsilon(\mathbf{v}) - \mathcal{J}_\varepsilon(\mathbf{u}^t) + \sum_{i=1}^{n} \frac{[e_i(\mathbf{v}) - e_i(\mathbf{u}^t)]^2}{e_i(\mathbf{u}^t)} &< \mathcal{J}_\varepsilon(\mathbf{u}^t) \\ \Leftrightarrow 2\mathcal{J}_\varepsilon(\mathbf{v}) + \sum_{i=1}^{n} \frac{[e_i(\mathbf{v}) - e_i(\mathbf{u}^t)]^2}{e_i(\mathbf{u}^t)} &< 2\mathcal{J}_\varepsilon(\mathbf{u}^t). \end{aligned}$$

Since

$$\sum_{i=1}^{n} \frac{[e_i(\mathbf{v}) - e_i(\mathbf{u}^t)]^2}{e_i(\mathbf{u}^t)} \geq 0,$$

it follows that

$$2\mathcal{J}(\mathbf{v}) < 2\mathcal{J}(\mathbf{u}^t) \Rightarrow \mathcal{J}(\mathbf{v}) < \mathcal{J}(\mathbf{u}^t).$$

□

Valkonen [381] has extended the above proof to encompass the missing value treatment and acceleration step. Hence, also the next theorem is proved.

Theorem 6.3.2. *If* $\omega \in]1, 2]$, *then* $\mathcal{J}_\varepsilon(\mathbf{u}^{t+1}) < \mathcal{J}_\varepsilon(\mathbf{u}^t)$.

6.3.2 SOR-accelerated Weiszfeld algorithm with inlier trimming and missing data treatment

This approach is based on the results of Chapter 4 indicating that the well-known "curse of dimensionality" problem is actually a beneficial feature for the inlier problem because of the increased sparsity of the data, which leads to the decreased probability of inliers. This algorithm is called ASSOR and it corresponds to the one presented in Section 6.3.1, but the well-defined subgradient defined in (39) for $\mathbf{u} \neq \mathbf{x}_i$ is used. The non-smoothness is handled by ignoring the current inliers at each iteration. A somewhat similar principle, called Winzorizing, is also applied by Brown et al. [55]. However, they utilized the quasi-Newton optimization algorithm in the function minimization. In order to realize the idea, the neighborhood ϕ must be defined for the solution. Hence, if $\mathbf{x}_i \in B(\mathbf{u}, \phi)$ for some $i \in \{i, \ldots, n\}$, then these points are discarded from the current solution of $\mathbf{v}$. The treatment of missing data follows the principles of the above SOR-algorithm. The optimality condition is now given as

$$\sum_{i:\mathbf{x}_i \notin B(\mathbf{u},\phi)} \frac{\mathbf{P}_i(\mathbf{u} - \mathbf{x}_i)}{\sqrt{\|\mathbf{P}_i(\mathbf{u} - \mathbf{x}_i)\|^2}} = 0. \tag{57}$$

The ASSOR algorithm is as follows:

Algorithm 6.3.2. ASSOR

Step 1. Initialize $\mathbf{u}$ and set ϕ and ω.

Step 2. Discard all $\mathbf{x}_i \in B(\mathbf{u}, \phi)$ (The number of remaining data points is denoted by m).

Step 3. Solve α_i^ts for $i \in \{1, .., m\}$ using (53).

Step 4. Solve the basic iterate $\mathbf{v}$ using the remaining data and (54).

Step 5. Accelerate $\mathbf{u}$ using (55).

Step 6. If the stopping criterion is satisfied then stop, else return to step 2.

6.4 Numerical and statistical experiments

Accuracy, reliability, and computational requirements including the scalability issues for the implemented algorithms are compared through the following numerical experiments. In order to ensure the adequacy of the results, a number of synthetic data sets, involving varying numbers of data points, dimensions, and shapes, were generated and utilized. The sensitivity to initial conditions was evaluated by running each algorithm from several different starting points.

From a statistical point of view, the properties, such as bias and efficiency of the consequent spatial median estimators, are validated through statistical experiments. As a part of the validation, the performance of the chosen missing data treatment is also inspected. The ultimate goal of the experiments is to find out which methods and formulations produce fast and consistent solutions with high statistical efficiency for the problem of the spatial median.

Because of the missing closed-form formulation and consequent absence of exact analytical solutions for the spatial median, the reference values for the experiments were obtained by using the Nelder-Mead algorithm and extremely strict stopping criterion ($\max_{i \in \{2,\ldots,n+1\}} \|\mathbf{x}_1 - \mathbf{x}_i\|_\infty < 10^{-12}$). The starting points for the NM algorithm were approximated by CG method. This was done, because the convergence is not guaranteed for NM from arbitrary points. Moreover, the exact solutions are obtained in shorter time in this way. The reference results are presented in Table 15. Scalability of the SOR-based algorithms to high dimensional problems is also validated.

6.4.1 Implementation of the algorithms and test settings

MATLAB software was used as a test environment. Except Nelder-Mead, the algorithms were self-implemented. In the case of Nelder-Mead, a MATLAB Toolbox implementation was employed. The Polak-Ribiere variant was the chosen CG method. The self-implemented GS algorithm was utilized in the one dimensional line search of CG. Because different gradient approximations were tested with the CG method, acronyms CG1 and CG2 are used in the text and tables to indicate the applied formulation. CG1 refers to the approximated formula in (48) and CG2 to the one given by (50).

The following parameters were used in the experiments.

CG1NM GS line search length was $s_{int} = 1$. Stopping criteria in GS and CG: $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-3}$ and $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-2}$, respectively. The maximum iteration counts in GS and CG: 50000 and 2000, respectively. Approximation parameter in (48): $\varepsilon = 1.49 \times 10^{-8}$. Stopping criterion in NM: $\max_{i \in \{2,\ldots,n+1\}} \|\mathbf{x}_1 - \mathbf{x}_i\|_\infty < 10^{-6}$.

NM Parameter values equal the ones in CG1NM.

CG1/CG2 GS line search length equals CG1NM. Stopping criteria in GS and CG: $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-8}$ and $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-6}$, respectively. The maximum iteration counts in GS and CG: 50000 and 1000, respectively. The approximation parameter in (48): $\varepsilon = 1.49 \times 10^{-8}$.

SOR/ASSOR The over-relaxation parameter ω was tuned according to experiments on the test data sets (see, App. 2-3). The results show that the best value of ω varies significantly with respect to the data sets. Therefore, a compromise $\omega = 1.5$ was used. Stopping criteria: $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-6}$. Maximum iteration count: 500.

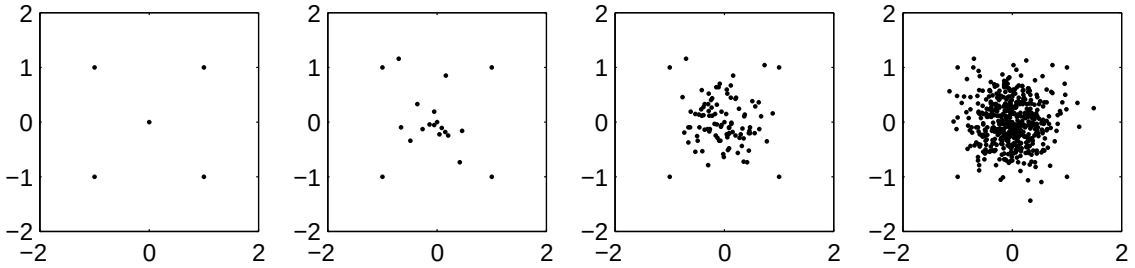


FIGURE 24 2D plots of the test data sets 1-4.

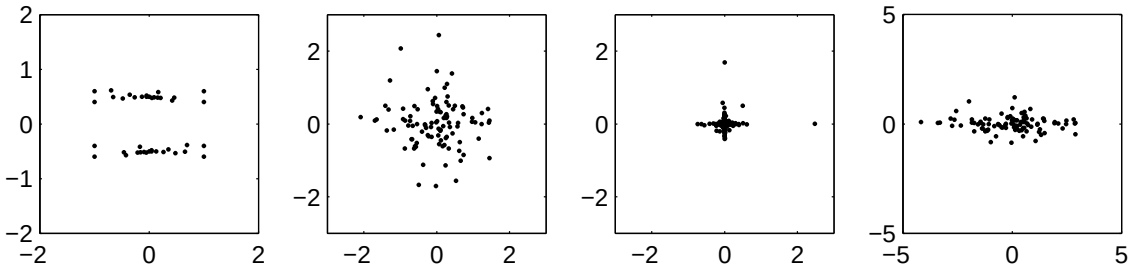


FIGURE 25 2D plots of the test data sets 5-8.

Modified Weiszfeld Stopping criteria: $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-6}$. Maximum number of iterations: 500.

The accuracy is evaluated according to error that is defined with respect to the corresponding reference value as (see Table 15):

$$e(\mathcal{J}(\mathbf{u}^*)) = \mathcal{J}(\mathbf{u}^*) - \mathcal{J}(\mathbf{u}^{ref}),$$

where $\mathbf{u}^*$ is the solution obtained by one of the experimented methods and $\mathcal{J}(\mathbf{u}^{ref})$ is the reference result obtained using NM with a strict stopping criterion. Correspondingly, the displacement error is defined by

$$e(\mathbf{u}^*) = \|\mathbf{u}^* - \mathbf{u}^{ref}\|_\infty.$$

6.4.2 Synthetic data sets

The first numerical tests were performed on eight two-dimensional data sets. The data sets were generated manually or by random sampling from the normal or Laplace distribution (see Figures 24 and 25). Note that instead of the actual statistical parameter values, the numerical accuracy of the solution is of concern in these experiments. Hence, the exact parameters of the underlying distributions are ignored here. In the case of data set 1, the exact solution for (38) is trivial and the error of the numerical solution can be computed precisely.

Starting points

Let us denote by $\mathbf{X}$ any of the experiment data sets. In order to test the reliability under different conditions, four different starting points were used on each data set. The following points were utilized:

- The sample mean of the data set.
- A data point $\mathbf{x}_i$ such that $\mathbf{x}_i \in \mathbf{X}$.
- An arbitrary point $\mathbf{x}$ such that $\mathbf{x} \notin conv(\mathbf{X})$.
- An arbitrary point $\mathbf{x}$ such that $\mathbf{x} \in conv(\mathbf{X})$.

6.4.3 Comparison of the results

In this section, accuracy, efficiency, and scalability of the algorithms are compared and discussed from a computational point of view.

Accuracy of algorithms on bivariate data sets

Figures 26 and 27 present trajectories of some interesting test cases. Each starting point is pointed by an arrow and the number of a test run. The trajectories are marked by dashed lines and their ending points by squares. The results show that CG1NM, CG1, Modified Weiszfeld, SOR, and ASSOR produce precise solutions with respect to the reference results (cf. Tables 16–18). The displacement errors were, at worst, approximately 10^{-4}, which is clearly a sufficient result. Data set 2 was the most problematic for CG1, SOR, and ASSOR.

The CG2 method gives inaccurate solutions. It seems that the smooth formulation of the problem leads to such rough approximations that the optimal solutions can not be attained by the CG2 method. Two samples of the trajectories illustrating the progress of CG2 can be found in Figures 26 and 27. Particularly the solutions of CG2 in Figure 27 are badly displaced for being the representative points of the data set.

Another problem concerning the CG2 method is presented in Figure 26. One can see that although the approximately optimal solution is attained, CG2 makes ”zigzag” steps that slow down the convergence speed. While ”zigzag”-steps are typical for the steepest descent method, they should be avoided by using the CG method. The better behavior by CG1 is shown by Figures 26 and 27. Results in Table 17 show that when initialized with starting points two and three, CG2 needs approximately 80 function evaluations more than CG1 for solving the problem on data set 1. On this basis, CG2 is a slow, unreliable, and inaccurate solver for the spatial median estimate.

NM method worked well except for the last run on data set 5, where the obtained solution was inaccurate in comparison to CG1NM, CG1, SOR, and ASSOR. In Figure 25 one can see that data set 5 is a kind of split set. The inaccurate convergence of NM may be due to initialization in this case as the difference between NM and CG1NM is in the initialization. Practically speaking, in the case of CG1NM, the NM algorithm is initialized with an approximate solution by CG1. Hence, the inaccuracy in NM solution is in line with the general assumption which says that the Nelder-Mead method is not guaranteed to converge from all arbitrary starting points (cf. Section 4.2.2), although it should converge relatively fast in the close neighborhood of the solutions. Hence, the errors on data set 5

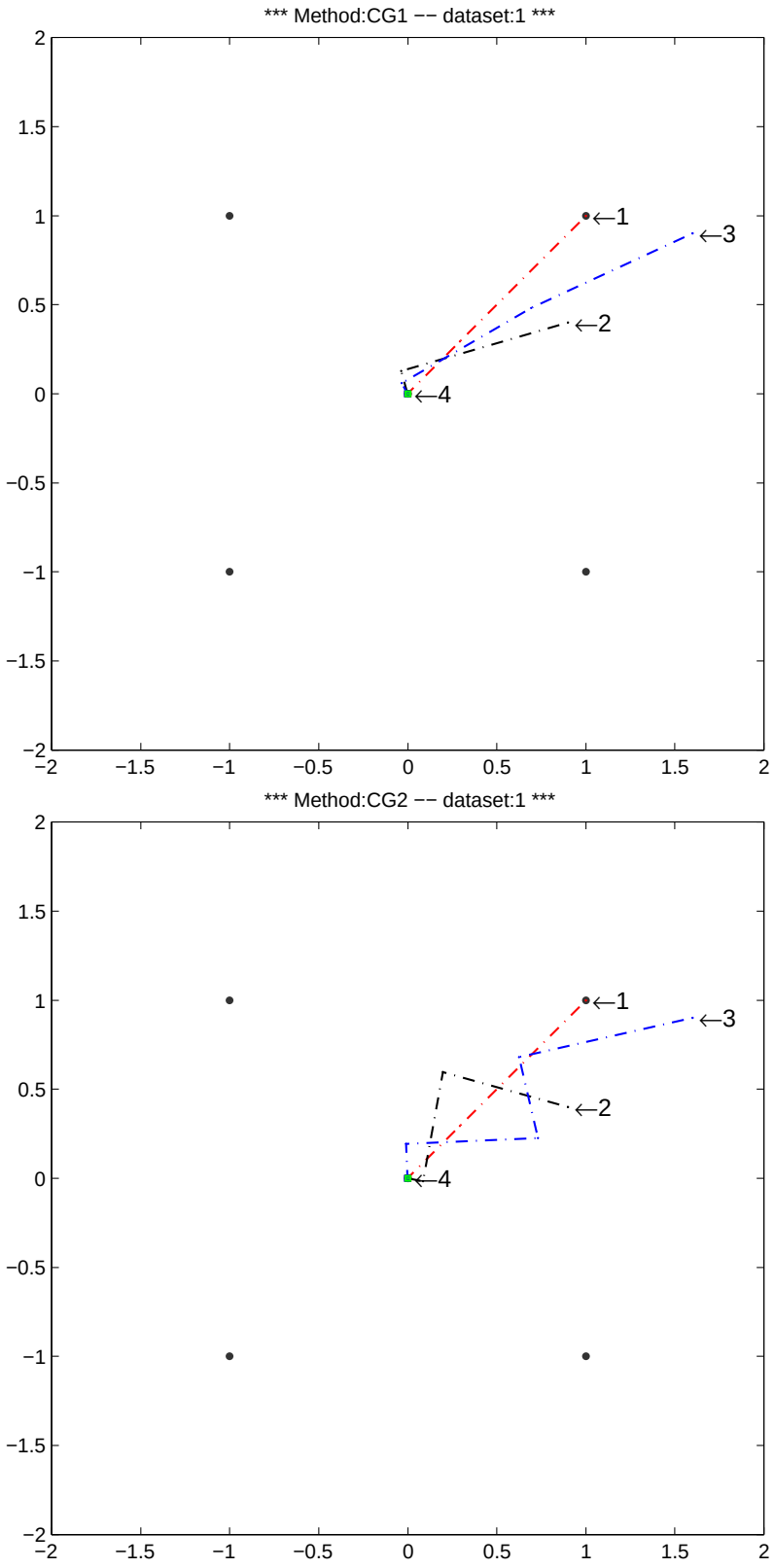


FIGURE 26 Trajectories of CG1 (top) and CG2 (bottom) methods on data 1.

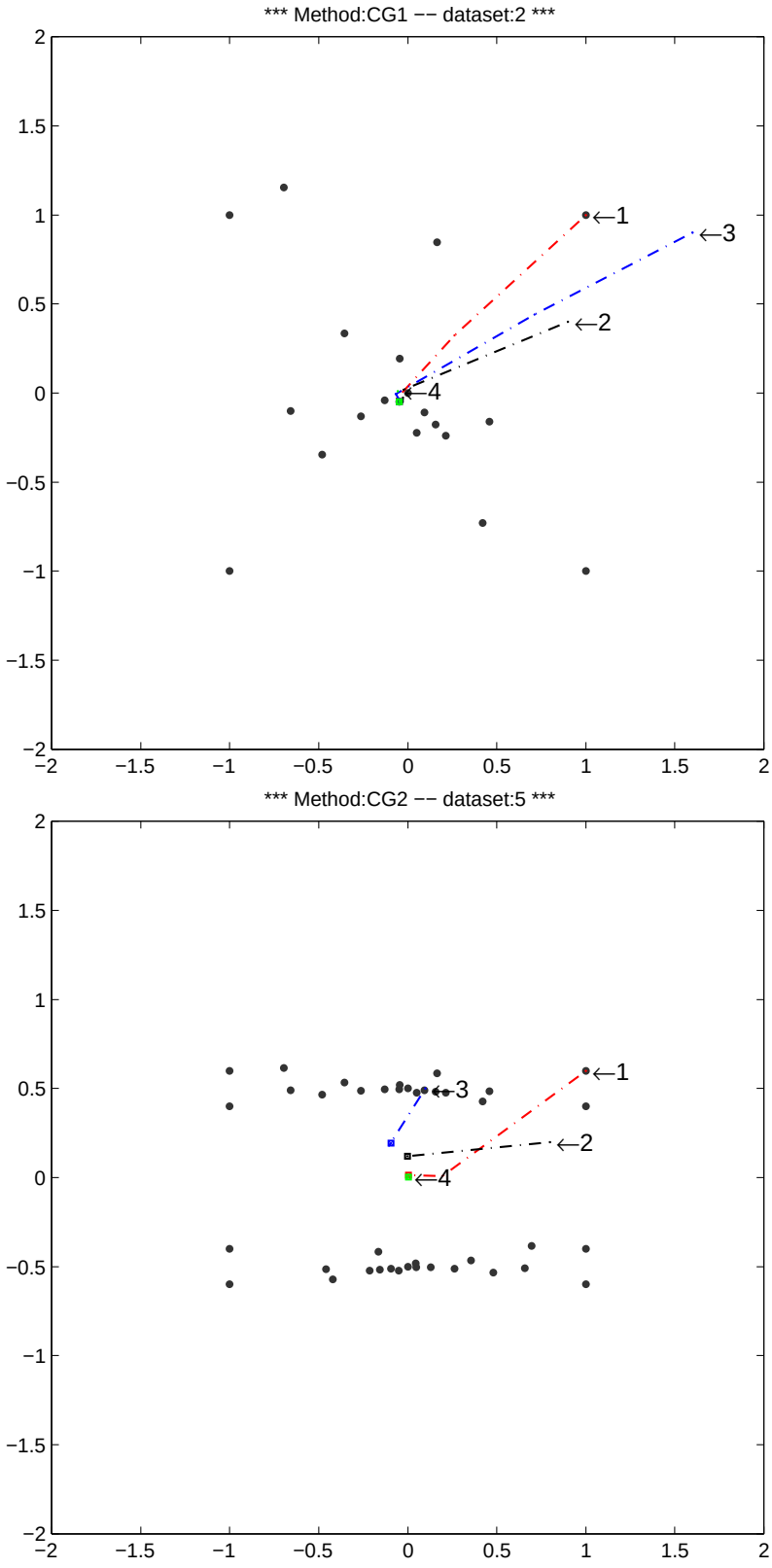


FIGURE 27 Trajectories of CG1 (top) and CG2 (bottom) methods on data 2 and 5, respectively.

warn us, at least on this particular type of data set, about the sensitivity of the simplex based methods to initial conditions. It is good to note that otherwise its solutions are consistent with the ones obtained by CG1NM.

As a remark from the DM aspect, it should be remembered that although data set 5 contains two quite clearly separated bunches of data, it can also be viewed as a single cluster. This is due to the ambiguities concerning the 'true' number of clusters that is a resolution dependent question (see, Chapter 3). For example, the data may be interpreted as one coherent cluster if the other clusters are distant to this particular cluster. Thus, sometimes one might need to compute estimates for such clusters and this gives a rationale to do experiments on this kinds of divided data.

It should be noted that the stopping criterion in the simplex-based Nelder-Mead optimization is not exactly comparable with the ones used in the other methods. However, as a whole, the results of these experiments show that regarding the accuracy of the obtained solutions, iterative SOR, conjugate gradient CG1, and the simplex method NM, provided that it is cautiously initialized, are precise algorithms for solving the problem of the spatial median.

Computational requirements

Because the basic principles in the considered algorithms differ, computational requirements can be analyzed only approximately. In order to approximate the costs, the number of the cost function evaluations are counted for the NM1CG and CG1 algorithm, and the number of iterations for the SOR and ASSOR solvers.

It is clear that more vector operations of the order $\mathcal{O}(p)$ ($\mathbf{u} \in \mathbb{R}^p$) are needed during the computation by CG and NM than by the simpler SOR-based methods. However, here we follow the main concerns of this thesis and analyze the results from the DM aspect using the more usual case when $n > p$ and the loops over the data dominate the computational cost. Hence, it is assumed that approximate but also adequate knowledge about CPU costs is obtained by comparing the aforementioned measures.

For evaluating the cost function given in (38), one pass through the whole data set is needed and means $\mathcal{O}(n)$ time complexity as the worst case. One SOR iteration approximates to two evaluations of the cost function as the whole data set must be passed through twice for one iteration. The first pass gives the solution of (53) and the second of (54), which then lead to $\mathcal{O}(2n)$ worst case time complexity. The worst case time complexity of ASSOR is approximately $\mathcal{O}(3n)$, which approximates three evaluations of the cost function. One more iteration compared to SOR is required for finding and pruning the inliers.

Comparison of CG1, CG1NM, and SOR-based algorithms Tables 16 and 17 show that CG needs 3.87 times more function evaluations than CG1NM method. This underpins the common arguments about the fast convergence of the NM algorithm in the neighborhood of the optimal solution. The solutions obtained by CG1 methods are slightly more precise than the ones obtained by CG1NM. This

may also be a consequence of the different stopping criteria used in the simplex and gradient based methods. A practical NM stopping criterion is based on the size of the simplex, whereas in gradient based methods, it is usual to evaluate the change of the solution. This gives a chance to use a somewhat looser stopping criterion for CG1, which terminates it earlier and thereby leads to a reduced number of cost function evaluations. However, if the number of the function evaluations accomplished by CG1NM is split between CG1 and NM parts, one can see that CG1 needs many more function evaluations even though it is used only in the initialization part of CG1NM with the clearly looser stopping criterion. Hence, due to the inherent properties of the "gradient-free" NM algorithm for solving non-smooth optimization problems and, moreover, the smaller number of the cost function evaluations, CG1NM is preferred to CG1. Therefore, CG1NM will next be compared to the SOR-based algorithms.

According to the above interpretation, the computational cost of one SOR iteration corresponds to two evaluations of the cost function. In the case of ASSOR, the same ratio is three. Let us first compare the requirements by SOR and CG1NM. By applying the above ratio number to the results presented in Tables 16-18, the average cost by CG1NM is six times greater than by SOR. When the results by ASSOR (see Table 18) are compared to the results by CG1NM, the CG1NM seems to be four times more expensive on average. This leads to the approximating result that CG1NM needs six times more computation for solving the spatial median problem than SOR and, respectively, four times more when compared to ASSOR.

It should also be noticed that in addition to the cost function evaluations, the NM algorithm makes, for instance, sorting operations to the data in every iteration, which requires additional $\mathcal{O}(n \log n)$ computing, especially on huge DM data sets.

Comparison of SOR-based algorithms to the modified Weiszfeld algorithm The SOR-based algorithms are also compared with the non-accelerated modified Weiszfeld algorithm. The results do not show significant differences in accuracy of the solutions. The number of iterations needed by the modified Weiszfeld algorithm is approximately 1.5 times greater than by SOR and ASSOR. Because the SOR iteration is also simpler and faster than the modified Weiszfeld iteration, both SOR and ASSOR are preferred to the modified Weiszfeld algorithm. On the other hand, the modified Weiszfeld algorithm can also be accelerated, which seems to produce enhancements similar to the SOR and ASSOR methods. Results for the accelerated versions of modified Weiszfeld algorithms are presented, e.g., in [387, 90]. The proposed methods do not provide any treatment or results for missing or high dimensional data.

The above results given in this chapter show that the SOR-type of algorithms are fast and accurate solvers for the spatial median (38). Therefore, they are good candidates of fast and reliable algorithms for computation of robust cluster location parameters in huge, noisy, and incomplete data sets. In order to know more about the large-scale behavior, the scalability of SOR and ASSOR to

high dimensional data mining problems will be evaluated.

Before moving on to do further analysis about scalability to the high dimensions, it is good to analyze the differences of the proposed SOR-based algorithms. It is obvious that the computational requirements of the ASSOR algorithm are greater than those of SOR because the inliers are pruned at each iteration. On the other hand, if a sufficient number of data points are removed as inliers, it follows that the evaluation of (53) and (54) takes less time due to the reduced amount of data. Thereby, it is clear that on a certain, even though perhaps rare, type of data sets, ASSOR may need less computation than SOR. As this kind of data sets may however occur in the DM context, a short careful discussion is in order.

DM tasks focus on data sets that are too large for the fast memory of PCs. Computation on such large data sets requires a lot of hard-disk accesses, because all the data can not be loaded at the same time into the fast main memory. This means that by pruning the data points the number of hard-disk accesses during the evaluation of (53) and (54) might be reduced as a result. On the other hand, it is known that DM usually considers high dimensional data sets that are likely to be less dense than lower dimensional data sets. This decreases the chance of inliers and, therefore, the speed gain achieved by pruning of the data points may become negligible. Furthermore, the characteristics of data depend on the preprocessing and data transformation operations, particularly on dimension reduction and feature selection, that change the dimension before the computation takes place (e.g., the dimension reduction shortens distances between the data points). Hence, it is difficult to give a unique guideline for SOR and ASSOR. Nevertheless, the most important issues to be taken into account are the amount of available memory and data, and the number of dimensions that are used in the computation.

Scalability of algorithms to high dimensions

The scalability to the large-scale problems was measured by using eight high dimensional data sets (chosen dimensions were $\mathbb{R}^8$, $\mathbb{R}^{16}$, $\mathbb{R}^{32}$, and $\mathbb{R}^{64}$). The data sets were generated by duplicating the data sets 5 and 6 (see Figure 25). CG1NM was again used as the reference solver. The stopping criteria were the same as in $\mathbb{R}^2$ experiments.

Table 19 shows that the simplex-based NM algorithm does not scale to high dimensions. The solution involves errors even though the maximum number of iterations were exceeded in some cases. The CG1NM algorithm combination seems to result in quite precise estimates, but the amount of the function evaluations increases significantly due to the high dimensions. CG1 (see Table 20) seems to be more efficient, but equally precise to CG1NM regardless of the number of dimensions. It is interesting that the solutions obtained by CG1 are even more accurate than the reference solutions. As in the $\mathbb{R}^2$ experiments, CG2 produces very imprecise results. Figure 28 illustrates the remarkable difference between the classic optimization approach and iterative methods. Iterative methods are in the sense of computational costs tolerant against increasing dimensions. Accord-

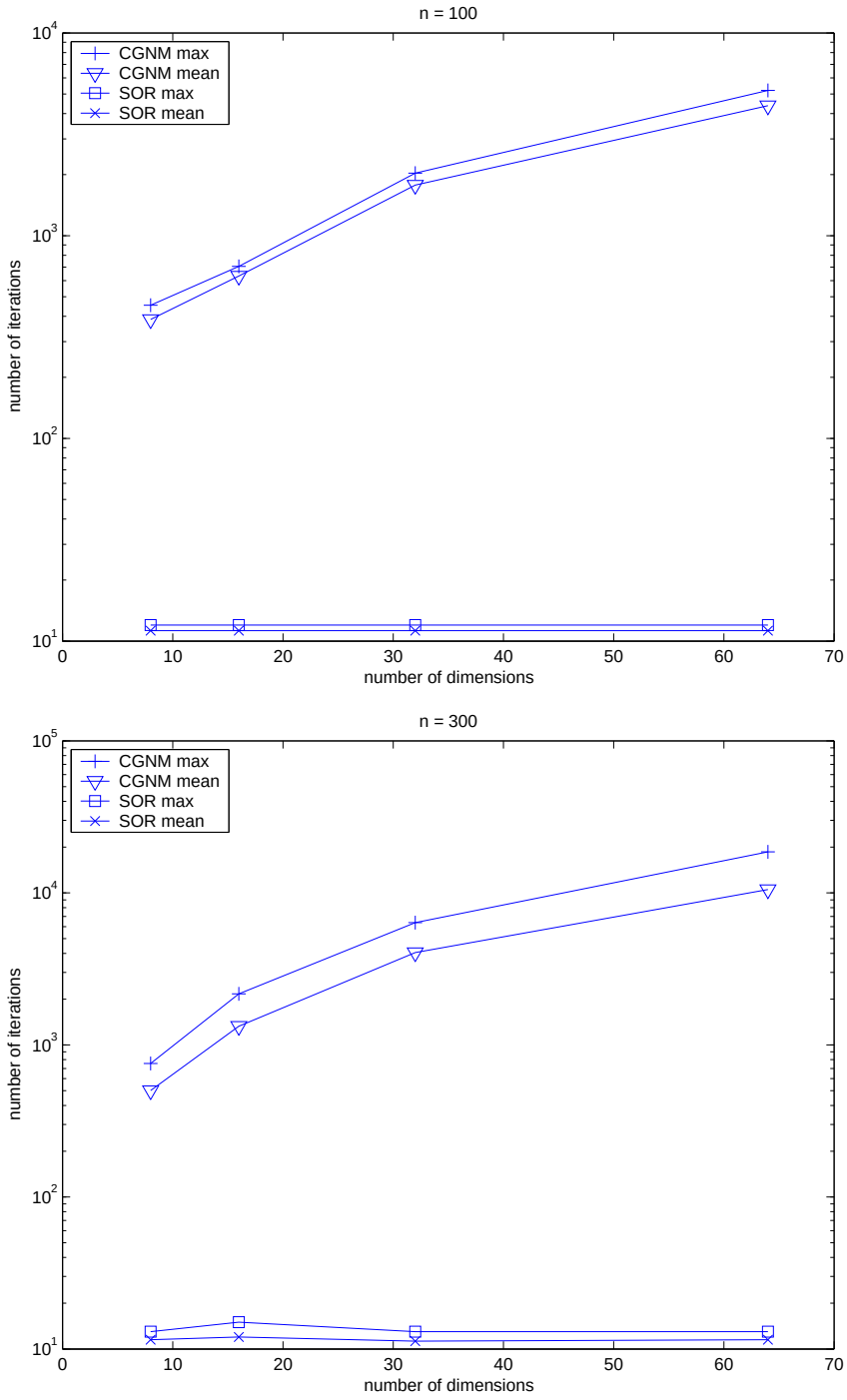


FIGURE 28 Scalability (top: $n = 100$ and bottom: $n = 300$) of the CG1NM and SOR methods to the high dimensional problems.

ing to the numerical experiments, CG1 seems to be the best solver of the classical optimization methods for the spatial median problem on high dimensional data (see Tables 19 and 20).

TABLE 2 Summary of the results for NM, CG1NM and CG on the bivariate test data sets (See Figures 24 and 25). "#CG", "#NM" and "total" are the numbers of function evaluations taken by CG, NM and the complete algorithms, respectively.

	NM			CG1NM				CG			
	$e(\mathcal{J}(\mathbf{u}^*))$	total	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	#CG	#NM	total	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	total	$e(\mathbf{u}^*)$
min	0.00e+00	37	0.00e+00	0.00e+00	4	37	41	0.00e+00	0.00e+00	4	0.00e+00
max	4.35e-04	112	4.34e-03	5.37e-07	158	76	211	1.29e-06	4.23e-07	1583	1.63e-06
mean	1.36e-05	104	1.36e-04	7.45e-08	95	52	147	3.78e-07	3.31e-08	583	1.77e-07
median	1.48e-11	111	2.82e-07	1.34e-11	96	51	147	2.98e-07	2.35e-13	477	3.42e-08

TABLE 3 Summary of the results for the modified Weiszfeld, SOR and ASSOR on the bivariate test data sets (See Figures 24 and 25). "it" is the number of iterations taken by an algorithm.

	MW			SOR			ASSOR		
data	$e(\mathcal{J}(\mathbf{u}^*))$	it	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	it	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	it	$e(\mathbf{u}^*)$
min	0.00e+00	1	0.00e+00	0.00e+00	1	0.00e+00	0.00e+00	1	0.00e+00
max	7.36e-07	41	3.08e-06	3.42e-05	26	1.42e-04	4.57e-07	26	1.49e-06
mean	8.93e-08	21	1.31e-06	4.31e-06	14	1.84e-05	6.61e-08	14	5.76e-07
median	9.93e-11	19	1.16e-06	1.51e-11	13	3.88e-07	1.81e-11	12	3.99e-07

Discussion

The summary of the results on the bivariate test data sets is presented in Tables 2 and 3. The results show that the iterative SOR-based algorithms clearly outperform the classical optimization approach. The precision of the results is the same and the scalability is superior to any of the classical optimization methods used. The number of iterations needed by the SOR-based methods seems to be independent of the number of dimensions. Moreover, it seems that the effect of the over-relaxation parameter ω is constant for the different number of dimensions. The obtained results, especially the scalability of the algorithms, are very remarkable and encouraging from the DM point of view.

6.5 Statistical experiments

After presenting the results from the computational point of view, the focus is turned to the statistical issues. Some testing for statistical properties were performed, and the results will be commented here. Basic properties for the spatial median estimator are tested and discussed, e.g., in [54, 92]. It is commonly known that robustness is often obtained at the cost of basic properties, such as unbiasedness, consistency, and efficiency[2] (cf. requirements in Section 4.4.3 and, see also, general statistical results on robust estimators [276]). Algorithms that are devel-

[2] NOTE: This means efficiency in a statistical sense.

oped for solving computationally unstable and expensive problems, such as the spatial median, may finally produce inconsistent estimates due to the approximations that must be done for cutting down the computation costs, for avoiding non-differentiability and extremely small numbers, and so on. Hence, the theoretically promising properties may become unachievable in real-world applications. In order to assure that the fast SOR-based spatial median estimators retain the statistical properties, a couple of statistical experiments were performed. Because the algorithms are generalized to the missing data cases using the available case strategy, the experiments were also performed on incomplete data sets. The experiments are a kind of Monte Carlo test for several sample sizes. The samples were drawn from the symmetric spherical multivariate normal and coordinate-wise independent Laplace distributions with the symmetry points (coordinate-wise mean/median= $\mathbf{0}$) at the origin and unit variance/scatter.

6.5.1 Consistency

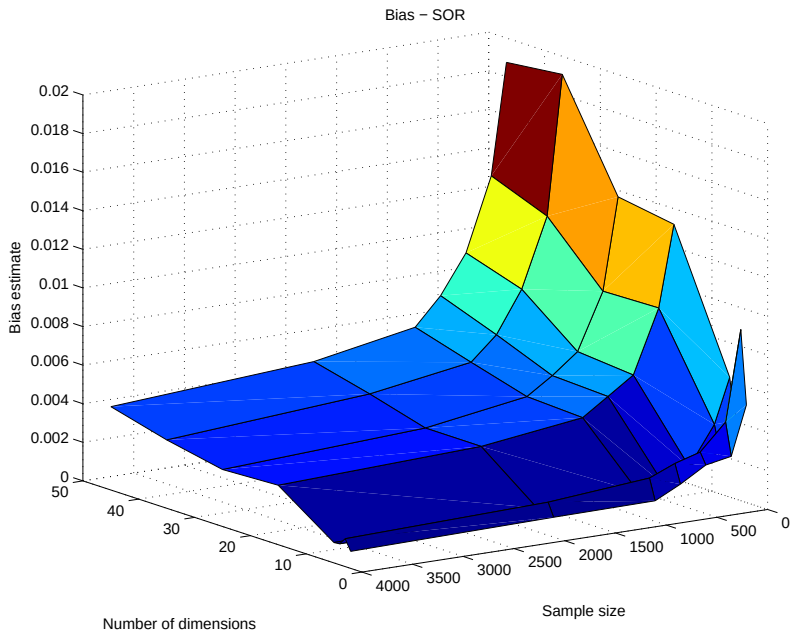


FIGURE 29 Estimated bias of the SOR estimator.

In order to analyze the consistency of the implemented spatial median estimators, the asymptotic behavior of the bias was estimated in a number of dimensions. For a consistent estimator, the bias approaches zero as $n \to \infty$. The estimator bias was measured as the average Euclidean distance between the estimates and the true parameter value of the generating distribution as

$$bias = \frac{1}{t} \sum_{i=1}^{t} \| T(\mathbf{x}_1, \ldots, \mathbf{x}_n) - \boldsymbol{\mu} \|_2,$$

where $\boldsymbol{\mu}$ is the true parameter of the sampling distribution and T is the tested estimator. In all tests the distribution is centered to the origin, hence $\boldsymbol{\mu} = \mathbf{0}$. For es-

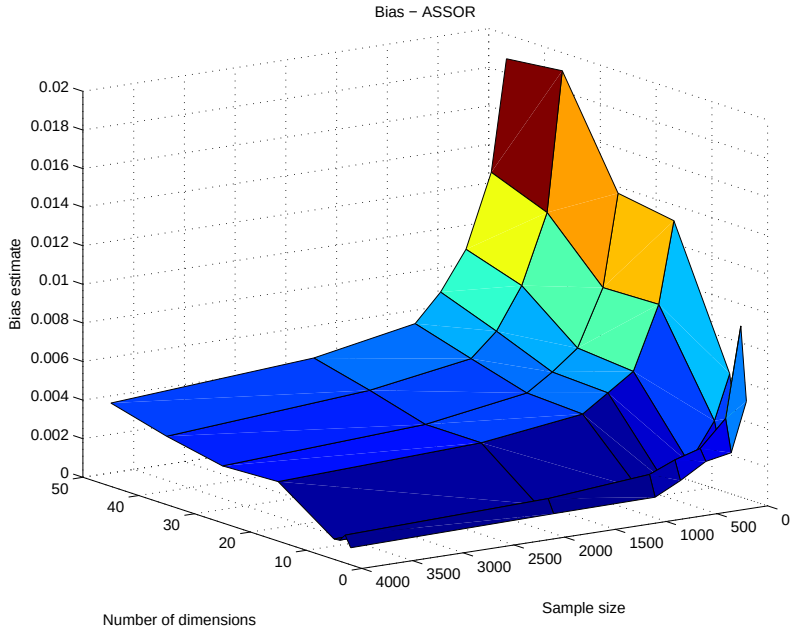


FIGURE 30 Estimated bias of the ASSOR estimator.

timating the asymptotic behavior of the bias, the experiments were accomplished on data sets of several sizes.

In the first tests, the bias was estimated for both SOR and ASSOR as the average of 1000 estimator biases on complete normal data that was sampled from $N_p(\mathbf{0}, \mathbf{1})$ distribution. The test dimensions were $p = \{2, 3, 4, 5, 15, 25, 35, 45\}$ and the size of the samples $n \in \{100, 250, 500, 750, 1000, 2000, 4000\}$. Figures 29 and 30 illustrate the results. Both of the estimators show consistent behavior, that is diminishing bias due to the sample size, in all the sample dimensions on complete normal data. The increased bias with respect to dimensions is simply explained by the fact that the Euclidean distance grows along with the number of dimensions.

The bias effect of the missing data treatment was also considered by estimating the bias on samples from which 15% or 40% of values were eliminated by MCAR mechanism. Two sampling distributions were used. The expected bias was computed as the average of 100 estimates for each case. Table 23 shows that both SOR and ASSOR estimators indicate consistency, because the estimated bias mitigates for both as the size of the sample grows. One can also see that the bias of the spatial median estimators behaves nearly equally with the sample mean bias estimates when the underlying distribution is Gaussian. In the case of Laplace distribution, the coordinate-wise sample median is naturally the least biased estimator, but the difference to other estimators is insignificant. Furthermore, the behavior of the bias estimates is approximately the same for the complete and incomplete data. As a whole, the results follow theoretical assumptions as all the estimators should be unbiased estimators of symmetric distributions. As a result, it can be concluded that neither the algorithmic approximations nor the missing data strategy cause bias to the spatial median estimates obtained by SOR and

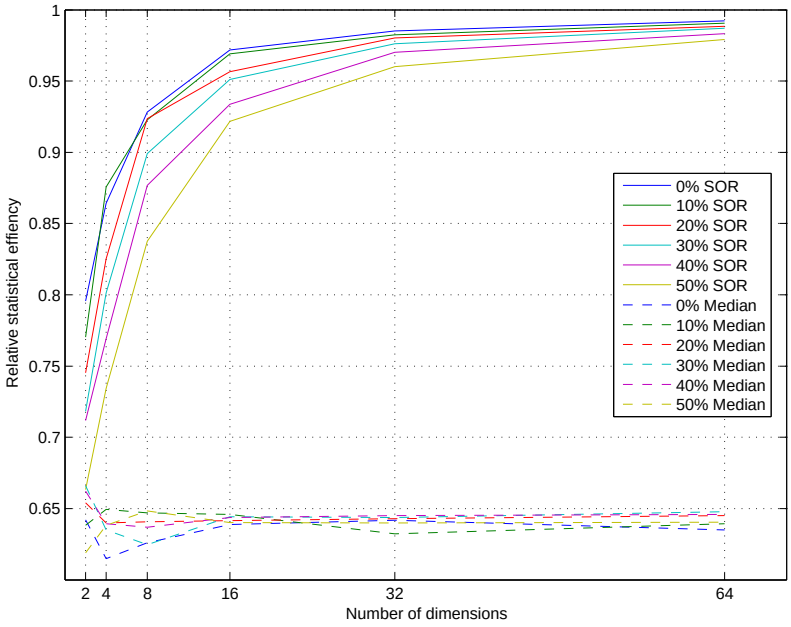


FIGURE 31 The relative statistical efficiency of the SOR type of spatial median estimator and the sample median with missing data treatment to the sample mean on 1000 samples of size of 200 from $N_p(\mathbf{0}, \mathbf{1})$ in the presence of different proportions of missing data. (SOR parameters: $\omega = 1.65$ and stopping criteria $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-6}$)

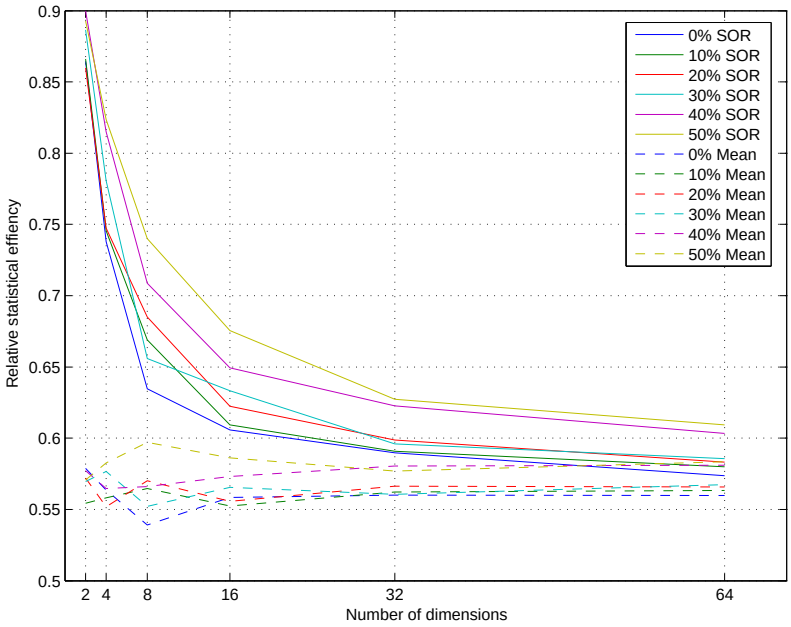


FIGURE 32 The relative statistical efficiency of the SOR type of spatial median estimator and the sample mean with missing data treatment to the sample median on 1000 samples of size of 200 from $L_p(\mathbf{0}, \mathbf{1})$ in the presence of different proportions of missing data. (SOR parameters: $\omega = 1.65$ and stopping criteria $\|\mathbf{u}^{t+1} - \mathbf{u}^t\|_\infty < 10^{-6}$)

ASSOR algorithms.

6.5.2 Efficiency on large-scale samples

The relative efficiency of the estimators was measured by estimating the average coordinate-wise sample variance on multivariate normal and Laplace-distributions. As the samples were drawn from symmetric distributions, the average coordinate-wise sample variance is a reasonable estimate of the estimator's variability for measuring its relative efficiency. The distributions were centered at the origin and scattered symmetrically with unit variance. The efficiency is expressed as a relative value with respect to the maximum likelihood estimator of the particular distribution. A somewhat similar approach is used, e.g., by Brown in [54]. He inspected the relative efficiency of the spatial median by comparing the variances in the direction of the principal components of the multivariate elliptical samples. The interesting question is the size of losses that the approximation and pruning implementations can cause to the estimators. Further, how does the chosen missing data strategy influence the efficiency of the estimators? The experiments were performed on several incomplete data sets as well.

The results are summarized in Table 22. At first, as explained in the theoretical part of this thesis, the coordinate-wise median is the most inefficient estimate on multivariate normal conditions. The results approximate well the theoretical value ($2/\pi \approx 0.64$). The behavior on normal conditions points out its univariate nature, since the growing number of dimensions does not provide any support for the estimates and their efficiency remains at a poor level. Because the spatial median coincides with the coordinate-wise median in the univariate case, the relative efficiency of the spatial median estimators is also weak in low dimensions. However, as the previous results (e.g., [54]) have shown, their efficiency should asymptotically approach the sample mean as the sample dimension grows. Actually, the SOR-based accelerated algorithms with missing data treatment seem to satisfy this well: the efficiency of the spatial median estimator and the sample mean should coincide if $p \to \infty$. This behavior for SOR-based implementations is illustrated in Figure 31 for a complete case and for several incomplete ones.

Table 4 presents a number that indicates the relative effect of the MCAR missing data to the efficiency of the SOR-algorithm with respect to the efficiency of the sample mean computed before the missing data was generated. The efficiency values are very closely related to the amount of missing data. The result is interesting, because it shows that the algorithmic realizations or the missing data technique used do not cause additional efficiency loss by themselves. It is important that the efficiency of any DM estimator is not collapsed by missing data, because missing data usually exists in real-world cases.

The variability of the estimators were also estimated on the symmetric coordinate-wise Laplace distribution. Also in this case, the results in Table 22 follow the theoretical assumptions about the position of the coordinate-wise median as the maximum likelihood estimator of the Laplace distribution. The relative efficiency of the sample mean varies between 53% and 59% independently of the number of dimensions or the amount of missing data.

The behavior of the spatial median estimators is equal to the earlier experi-

TABLE 4 Relative normal efficiency of the spatial median estimator in MCAR case with respect to the sample mean on complete data. Columns: dimension. Rows: % of missing data. Estimated from 10000 samples (n=100).

	2	4	8	16
0%	1.0000	1.0000	1.0000	1.0000
10%	0.9035	0.8985	0.9013	0.8988
20%	0.8005	0.7952	0.7989	0.7984
30%	0.6975	0.7026	0.6970	0.6977
40%	0.5993	0.5968	0.5975	0.5949
50%	0.5101	0.4972	0.5006	0.4937

ments under normal conditions, since their relative efficiency approach the sample mean as the number of dimensions grows (see Figure 32). As the relative amount of missing data is increased for the Laplace distributed samples, the relative efficiency of the spatial median and the sample mean show slight improvement.

6.5.3 Discussion

In general, the experiments show that the statistical assumptions are satisfied by the new modified spatial median implementations. The chosen missing data treatment seems to maintain the efficiency of the spatial median estimator, since the efficiency lost is comparable to the amount of missing data. In other words, the chosen missing data treatment does not cause additional efficiency loss to the spatial median estimators. These results follow and complete previous results, for example, by Brown [54] and satisfy the theoretical assumptions presented by the statisticians in the course of years since the sixties. The results also show that the developed algorithms and implementations do not lead to biased and inconsistent estimates. The asymptotic relative normal efficiency of the spatial median with respect to the number dimensions is also a significant result, since the usual data mining tasks handle very high dimensional data sets.

6.6 Conclusions

The goal of this chapter was to develop fast and reliable algorithm with missing data strategy for the robust estimator called spatial median. Several different formulations and methods were tested and compared both numerically and statistically in order to find the most consistent and fastest realizations. The convergence of the perturbed Weiszfeld iteration was considered. The implemented SOR-based estimators can be applied in many clustering methods, such as the

ones presented and initially tested in Chapter 5. Since the real-world data sets usually contain erroneous and missing values, a special missing data treatment was implemented. Hence, a new formulation for the problem of the spatial median taking into account missing values was given. Its influence on the statistical properties is investigated through the statistical experiments.

7 INITIALIZATION METHODS FOR CLUSTERING ALGORITHMS

One of the underlying problems concerning data clustering is the global non-uniqueness of the solution. This problem is due to non-convex nature of clustering problem, or its cost function itself. Non-convexity of a function means that it possesses multiple local minima. Therefore, the use of arbitrarily chosen initial points in a clustering problem often ends up with a locally optimal, but unsatisfactory, partition. Moreover, for a bad initial guess, an increased number of clustering iterations is usually required. On the other hand, finding the globally best partition by exhaustive search is not practical due to the huge number of different partitions even for small data sets (see Section 3.3).

Prototype-based partitioning algorithms, such as K-means [265] and K-spatialmedians [208], are particularly dependent on the initial cluster centers as they are based on the local-search principle and, thereby, converge to some locally best partition in the neighborhood of the initial points. Furthermore, they are often prone to so-called *dead clusters*, providing the initial points are poorly chosen. Dead cluster is a cluster that does not attract any data points upon convergence [182]. As it is unachievable to create a universal, data-independent, clustering method [220], it is not realistic to aim at developing a universal initialization method either. Perhaps for this reason the strategy of multiple repetitions with random initial points has remained as the *de-facto* method for the initialization [43].

Although the random initialization obtains the initial points extremely fast, the problem is that the number of iterations needed by the actual clustering algorithm grows with unsuccessful initialization. Hence, the multiple repetition from random points is perhaps a feasible solution for small data sets, but on large data sets, with hundreds or thousands of dimensions and, perhaps, thousands or millions of objects, it may take hours or even days to repeat the clustering algorithm. Moreover, one needs to define a criterion to choose the ”best” result from the solutions obtained starting from different random initial solutions. A more sophisticated way to solve the problem of multiple local minima is to apply some heuristic method to determine the initial conditions for a clustering algorithm.

Basically, various initialization methods have been developed for clustering algorithms, see, e.g., [9, 273, 315, 43, 218, 222, 6, 182, 119, 365, 140, 264]. However, only some of them have so far been developed or tested for DM purposes. Consequently, they have not been tested on DM type data sets, which are often large, incomplete, heterogeneous, and erroneous.

7.1 Basic methods for the cluster initialization problem

He et al. [182] classify the initialization methods into three classes: *random*, *distance optimization*, and *density estimation*. In the following, short descriptions about these basic methods are given.

7.1.1 Random initialization

Perhaps the most common, naive and *de-facto* method of the clustering initialization problem is the random initialization [9]. There are actually several ways to realize random initialization, but the basic idea is to simply initialize the cluster centers with random data points. This approach is also referred to as Monte Carlo codebook design by Gersho and Gray [140, p.359]. When the random initialization is used as an initialization strategy, the algorithm must be run several times in order to obtain enough conviction about the quality of the clustering.

By restricting the choice of the random data points, different variants for the random initialization are obtained. First of all, the initial cluster centers can be restricted to the given data points, that is $\{\mathbf{m}_k\}_{i=1}^{K} \subset \{\mathbf{x}_i\}_{i=1}^{n}$. One may also choose the initial centers arbitrarily using the range of the individual variables, in which case they will be unlikely to intersect with the individual data points and, hence, the chance that the initial points lie outside the convex hull of the data set remains. If all values, even outside the data range, were allowed to be chosen as the initial points, some of them might be so distant to the actual data cloud as to become unable to capture any data points and, consequently, dead clusters would unavoidably occur. Therefore, in order to avoid dead clusters and obtain fast convergence, it may be better to restrict the random initial points to the convex hull of the data set. A strategy, that is assumed to decrease the chance of dead clusters, is to slightly and randomly disturb the mean of the whole data set for a required number of times and use these points as the initial cluster centers [182]. This is supposed to give at least nearly even probability to all cluster centers of being selected as the closest prototype during the first clustering iterations.

When sequential clustering algorithms, such as the MacQueen's K-means algorithm [265], are used, one may initialize them by using the K first data points as the initial points. This is a practical approach, for example, in real-time applications that receive the data in sequence and prior information about its internal structure may not be available. For highly correlated data sets Gersho and Gray [140, p.359] suggest to use every Kth rather than the first K data points as initial

centers.

Since quality of the clustering solutions obtained with the random starting may vary a lot, some heuristics for the cluster initialization problem have been developed. In the following, some of the common approaches are presented more precisely.

7.1.2 Distance optimization methods

The main idea behind the distance optimization methods is to maximize the distances between the cluster centers beforehand. This is reasonable, since clustering criteria are often based on the minimization of intra-cluster distances and maximization of inter-cluster distances, which actually leads to a kind of multi-objective optimization problem [280]. However, many of the popular clustering algorithms, for example, K-means [265], K-medoids (or PAM) [220], or K-spatialmedians [208, 104], ignore the between-cluster distances. Hence, such clustering algorithms may benefit from taking the between-cluster distances into account as a part of the initialization.

Katsavoudinis et al. [218] propose a very fast distance-based initialization technique for the generalized Lloyd's iteration, which corresponds to K-means iteration (c.f. Section 3.3.2). Following the convention by Al-Daoud and Roberts [6], the method will be termed here as KKZ. The main principle of the KKZ algorithm is to take the data point which differs most from all the existing prototypes as the next prototype and repeat this process until the desired number of prototypes is found. An interesting detail is that no input parameters are needed, which is especially an attractive feature from the DM point of view. The method has produced very competitive results in the experiments presented in [6] and [182].

The detailed algorithm is given next.

Algorithm 7.1.1. KKZ

Input parameters: Data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{n}$, where $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, \ldots, n$, and the desired number of clusters K.

Output parameters: The initial prototypes $\{\mathbf{m}_1, \ldots, \mathbf{m}_K\}$.

Step 1. Set $k = 1$. Compute the norm (e.g., Euclidean norm) for all data points $\mathbf{x}_i$ $(i = 1, \ldots, n)$. The first prototype $\mathbf{m}_1$ is the one with the largest norm

$$\mathbf{m}_1 = \arg\max_{\mathbf{x}_i} \|\mathbf{x}_i\|.$$

Step 2. For all the remaining data points in $\mathbf{X}$, define the closest prototypes of the hitherto determined ones as

$$(\mathbf{c})_i = \arg\min_{j \in \{1,\ldots,k\}} \|\mathbf{m}_j - \mathbf{x}_i\|,$$

where $\mathbf{c} \in \mathbb{N}^n$ is a code-vector that contains the index of the closest prototype for every point. The next prototype is defined as

$$\mathbf{m}_{k+1} = \arg\max_{\mathbf{x}_i} \|\mathbf{m}_{(\mathbf{c})_i} - \mathbf{x}_i\|.$$

Set $k = k + 1$.

Step 3. If $k < K$ then repeat Step 2. Otherwise, stop.

The KKZ algorithm is very fast as the distances from the data points to the existing cluster centers need to be computed only with respect to the newest prototype. The computational complexity of KKZ is $\mathcal{O}(Knp)$, which is comparable with the K-means algorithm. A difficult problem is, however, the treatment of the missing data values that are typical in the DM context. Another problem is its sensitivity to outliers, which is illustrated in Figure 33. In this thesis, KKZ will be generalized to the missing data cases.

SCS (Simple Cluster-Seeking Algorithm) is another promising distance optimization method [182]. The results by SCS are actually nearly comparable to KKZ [372]. It was originally introduced as a clustering method, but later used also as an initialization method. From the DM point of view, the problem of SCS is the determination of a threshold parameter, since such prior knowledge is not expected to be available. Hence, this method will not be further considered in this thesis.

7.1.3 Density estimation method

Use of the density estimation strategy for the cluster initialization problem is based on the observation that random sampling yields information about the modes of a multivariate data set. The mode estimates can then be used as initial locations of the cluster centers. In the case of large data sets, it is more efficient to draw a number of small random subsets from the data and use them for finding high-density areas of the data for initialization. The sub-dataset clustering is an interesting approach in the sense that it also produces information about the number of clusters. Many clustering validation methods, such as resampling [336], stability-based [243, 291], and prediction-based methods [370, 95], are based on reproducibility of cluster assignments. Hence, the variability in the clusters found for the random sub-datasets can possibly be exploited in the determination of the number of clusters.

Cluster refinement algorithm

Bradley et al. [43] introduce a refinement procedure for the initialization problem. The method is referred to here by the acronym BF. It can be used for a wide class of clustering algorithms that assume the compact spherical shape of clusters. Although Bradley et al. talk about sub-samples, in this thesis these are called sub-datasets, and the terms data and dataset are used to replace the term sample.

The idea of the BF algorithm is to create a refined dataset by clustering a number of small random sub-datasets, and then cluster the refined dataset by using the cluster centers of the sub-datasets as initial points. As the final initial points are chosen the ones with the smallest distortion. BF takes as an input the number J of sub-datasets. Bradley et al. used $J = 10$ in their experiments. The same approach can be used for any other prototype-based clustering algorithm (e.g., [119]).

KMeansMod

For the sub-dataset clustering, a special KMeansMod algorithm is given that terminates when K non-empty clusters, denoted by $\{\mathcal{C}_k\}_{k=1}^{K}$, are found. The dead clusters are avoided by re-initializing the clusters having zero assignments of data points with the worst fitting data points and running the algorithm again. The worst fitting point is the one from which the distance to the closest cluster center is largest. After running the KmeansMod algorithm we have K non-empty clusters that are applied in the successive steps of the BF algorithm.

Algorithm 7.1.2. KMeansMod

Input parameters: Data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{n}$ where $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, \ldots, n$ and the number of clusters K.

Optional input parameters: Initial centers $\{\mathbf{m}_k\}_{i=1}^{K}$ where $\mathbf{m}_k \in \mathbb{R}^p$ for all $k = 1, \ldots, K$.

Output: Cluster centers $\{\mathbf{m}_k^*\}_{k=1}^{K}$ so that for all clusters $\mathcal{C}_k \subset \mathbf{X}$, it holds $|\mathcal{C}_k| \geq 1$ for all $k = 1, \ldots, K$.

Step 1. (*K-means clustering*) Using predetermined or random initial points, partition $\mathbf{X}$ into K clusters by the K-means algorithm. If provided, use the initial centers, or generate random initial points.

Step 2. (*Re-initialization*) If one or more clusters that capture zero data points are met, re-initialize with the worst fitting points, that is, the points that have the largest distance to their closest cluster centers. Go back to Step 1 and use the current cluster centers as the initial centers. If there are no empty clusters, then terminate.

When the data set is complete, that is no values are missing, the most unfitting point can be determined by using, e.g., the l_q-norms. But, in the presence of missing values, one should use a more general distance measure, such as the Gower's general distance measure [147] or the one given by (6). By standardizing the distances by the number of available variables, these measures make the distances between the points with different numbers of available values comparable.

Next the actual BF algorithm is described. At the beginning, a predetermined number of random sub-datasets of equal size are drawn from a given data

set $\mathbf{X}$. The probability of becoming selected into the sub-dataset is equal for each point in $\mathbf{X}$. The number and size of sub-datasets are denoted by J and $\tilde{n}$, respectively. If $\tilde{n}$ is large, more information is utilized, but more computation is also required. A good choice might be $\tilde{n} = n/J$. In order to have J set of cluster centers with non-empty clusters for X, all sub-datasets are clustered by the KmeansMod algorithm. The centers of the sub-dataset clusters constitute the refined data set, which is denoted by $\mathbf{X}_{ref}$. $\mathbf{X}_{ref}$ will be further clustered J times by using the obtained cluster centers as initial points in turn. The final set of initial points is the set of centers that have the smallest distortion with respect to the refined data set. The complete BF algorithm reads as follows:

Algorithm 7.1.3. BF

Input parameters: Data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{n}$ where $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, \ldots, n$, the number of clusters K, the number of sub-datasets J, and the size of the sub-datasets $\tilde{n}$

Output: Cluster centers $\mathbf{C}^* = \{\mathbf{m}_k^*\}_{k=1}^{K}$ so that for all clusters $\mathcal{C}_k \subset \mathbf{X}$, it holds $|\mathcal{C}_k| \geq 1$ for all $k = 1, \ldots, K$.

Step 1. Set $\mathbf{X}_{ref} = \phi$.

Step 2. For $i = 1, \ldots, J$.

Step 2.1. Draw $\tilde{n}$ data points with equal probability from X. Let us denote the resulting sub-dataset by $\tilde{\mathbf{X}}$.

Step 2.2. Find K cluster centers for $\tilde{\mathbf{X}}$ using Algorithm 7.1.2. Let us denote the centers by $\hat{\mathbf{C}}_i$.

Step 2.3. Extend the refined dataset as $\mathbf{X}_{ref} = \mathbf{X}_{ref} \cup \hat{\mathbf{C}}_i$.

Step 3. Set $\hat{\mathbf{C}}^r = \phi$.

Step 4. For $i = 1, \ldots, J$

Step 4.1. Take $\hat{\mathbf{C}}_i$ as the initial points and find K centers for $\mathbf{X}_{ref}$ by using the K-means algorithm. Let us denote the set of center points by $\hat{\mathbf{C}}_i^*$.

Step 5. The refined centers are $\mathbf{C}^* = \arg\min_{\hat{\mathbf{C}}_i^*} \mathit{Distortion}(\hat{\mathbf{C}}_i^*, \mathbf{X}_{ref})$.

Hence, the center points with the smaller *Distortion*, that is the smallest sum of the squared errors with respect to the refined data set $\mathbf{X}_{ref}$, are chosen as the initial points for the actual clustering method. K-means clustering solutions obtained with the BF initialization using $J = 10$ are compared to the ones obtained using random initial points in [43]. The results show that K-means finds the true clusters more efficiently when initialized with BF. Moreover, using the BF initialization the *K*-means algorithm converges in less iterations.

7.2 New methods

As far as the author of this thesis is aware, robust methods have not been utilized in the clustering initialization problems. It is well-known that the derivative of the cost function of the sample mean is unbounded. From this it follows that K means, i.e. the K cluster centers for data obtained by the K-means method, are unbounded as well. This makes the cluster centers extremely sensitive towards outlying values. In order to build robust algorithms, robust M-estimates with bounded derivatives are often used instead of the sample mean (e.g., [220, 208, 104, 207]).

Modifications for the existing methods are presented and aimed at making them robust against outliers and contamination. The methods are also generalized to handle missing data. In the case of the BF algorithm, this means that the sample mean estimates are replaced by the spatial median, and a particular strategy is applied to the missing data cases.

In order to diminish the tendency to empty or singleton clusters, a trimming procedure is applied. The trimming means, herein, removal of predetermined fraction of data points that are lying on the edge of a data cloud. Its use is based on the facts and ideas presented in [11], where the authors, García-Escudero and Gordaliza, state that the generalized K-means-type algorithms do not inherit robust properties from the bounded M-estimators. They show that the bounded derivative of the cost function of the M-estimator does not necessarily extend to the corresponding prototype-based clustering problem. As a result, an impartial trimming procedure is proposed for building a robust K-means method.

In this section, the trimming is performed only as part of the initialization problem, not in the actual clustering step. The overall goal is to find initial points where clustering algorithms converge to a local minima that optimally represent the high-density and compact areas of the data space. Before introducing the new methods, an imputation method, which is needed for the random and distance-optimization-based initialization, is presented.

7.2.1 Nearest-Neighbor imputation

Many real-world problems suffer from the fact that a part of data is missing. From this it follows that depending on the used methods, cluster centers $\mathbf{m}_k$ may contain empty values in some components. In order to proceed in a non-parametric fashion, the principles of the nearest-neighbor imputation are applied to fill in missing values according to the ideas of Batista et al. [27, 26, 25]. The basic idea is that after selecting data vector $\mathbf{x}_i$ ($i = 1, \ldots, n$), either randomly or by using some heuristics, the missing components of $\mathbf{x}_i$ are substituted with the existing ones of the most similar objects. Let $\mathcal{I} \subset \{1, \ldots, p\}$ be the index set for the missing components in a data vector $\mathbf{v}$. The imputation procedure is given by Algorithm 7.2.1.

Algorithm 7.2.1. Nearest-neighbor imputation

Input parameters: Data vector $\mathbf{v} \in \mathbb{R}^p$, data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, \ldots, n$.

Output parameters: Complete data vector $\mathbf{v}$.

Step 1. For $j = 1$ to $|\mathcal{I}|$

Step 1.a. (*Nearest-neighbor search*) Find $\hat{\mathbf{x}}$ by

$$\hat{\mathbf{x}} \leftarrow \arg\min_{i \in \{1,\ldots,n\}} \|\mathbf{x}_i - \mathbf{v}\|_G$$

subject to

$$(\mathbf{x}_i)_{\mathcal{I}(j)} \quad \text{is available for all } i.$$

Notation $\|\cdot\|_G$ refer to the general distance measure given by (6).

Step 1.b. (*Replacement*) Set $(\mathbf{v})_{\mathcal{I}(j)} = (\hat{\mathbf{x}})_{\mathcal{I}(j)}$.

The computational complexity of Algorithm 7.2.1 is $\mathcal{O}(np^2)$. Hence, it is somewhat sensitive to the number of variables, but this is not necessarily a serious problem, because for usual large DM data sets $n >> p$. The algorithm needs to be performed only once during the initialization.

7.2.2 robBF - Robust density-estimation initialization method

robBF follows the principles of the algorithm proposed by Bradley and Fayyad [43]. In order to make the BF method robust and include an efficient treatment for missing data, the sample mean is replaced with the spatial median and available data strategy is applied in all computation (see, Section 3.2.8). The spatial median is computed by using a fast SOR-based algorithm as described in Chapter 6.6 and [209]. In the presence of these changes, it is assumed that the initial points become more efficiently directed to the high-density areas of erroneous and incomplete data sets.

At first, the robust variant corresponding to the KmeansMod is given. As in the case of the basic K-means algorithm, KmeansMod can also be made robust by replacing the sample mean estimates with the spatial median. Handling of missing data values can be realized by projecting all computation to existing values as, for example, presented in [258, 208]. While the K-means-type algorithms may converge to a solution where one or more clusters capture no data the KmeansMod algorithm avoids these problems by re-initializing the empty clusters.

Algorithm 7.2.2. KSpatmedMod

Input parameters: Data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, \ldots, n$ and the number of clusters K.

Optional input parameters: Initial centers $\{\mathbf{m}_k\}_{i=1}^K$ where $\mathbf{m}_k \in \mathbb{R}^p$ for $k = 1, \ldots, K$.

Output: Cluster centers $\{\mathbf{m}_k^*\}_{k=1}^K$, so that for all clusters $\mathcal{C}_k \subset \mathbf{X}$, it holds $|\mathcal{C}_k| \geq 1$ for all $k = 1, \ldots, K$.

Step 1. (*K-means clustering*) Using predetermined or random initial points, partition $\mathbf{X}$ into K clusters by the K-spatialmedians algorithm. If provided, use the initial centers, or generate random initial points.

Step 2. (*Re-initialization*) If one or more clusters that capture zero data points are obtained, re-initialize with the worst fitting points, that is, the points that have the largest distance to their closest cluster centers. Go back to Step 1 and use the current cluster centers as the initial centers. If there are no empty clusters, then terminate.

As in Algorithm 7.1.2, the worst fitting points in Step 2. can be computed by using the general dissimilarity measure given by (6). The KSpatmedMod algorithm produces K non-empty clusters that are employed as part of the robust robBF and TrobBF algorithm, which are given in the following.

As the basic idea of robBF is exactly the same as in BF given by Algorithm 7.1.3, only the algorithm for robBF is given, and for further details the reader is directed to Section 7.1.3. The only difference between the BF and robBF methods is that the sample mean estimates are replaced by the spatial median. The complete robBF algorithm reads as follows:

Algorithm 7.2.3. robBF

Input parameters: Data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{n}$ where $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, \ldots, n$, the number of clusters K, the number of sub-datasets J and the size of the sub-datasets $\tilde{n}$

Output: K non-empty cluster centers $\{\mathbf{m}_k^*\}_{k=1}^{K}$.

Step 1. Set $\mathbf{X}_{ref} = \phi$.

Step 2. For $i = 1, \ldots, J$

Step 2.1. Draw $\tilde{n}$ data points with equal probability from X. Let us denote the resulting sub-dataset by $\tilde{\mathbf{X}}$.

Step 2.2. Find K cluster centers for $\tilde{\mathbf{X}}$ using Algorithm 7.2.2. Let us denote the centers by $\hat{\mathbf{C}}_i$.

Step 2.3. Extend the refined dataset as $\mathbf{X}_{ref} = \mathbf{X}_{ref} \cup \hat{\mathbf{C}}_i$.

Step 3. Set $\hat{\mathbf{C}}^r = \phi$.

Step 4. For $i = 1, \ldots, J$

Step 4.1. Take $\hat{\mathbf{C}}_i$ as the initial points and find K centers for $\mathbf{X}_{ref}$ by using the K-spatialmedians algorithm. Let us denote the set of center points by $\hat{\mathbf{C}}_i^*$.

Step 5. The refined centers are $\mathbf{C}^* = \arg\min_{\hat{\mathbf{C}}_i^*} Distortion(\hat{\mathbf{C}}_i^*, \mathbf{X}_{ref})$.

7.2.3 ModKKZ - distance-optimization-based initialization method for incomplete data

ModKKZ corresponds to the original KKZ method supplemented with a missing data treatment. The missing data sets extra-requirements for data selection and distance measures. As the usual Euclidean norm does not straightforwardly suit for distance comparisons if the data points contain different numbers of missing components, it is replaced by the general dissimilarity measure (6) in the ModKKZ. Furthermore, if the point chosen as the initial center contains missing values, it is filled in by the Nearest-neighbor imputation algorithm 7.2.1. In order to utilize most of the available information, the data will be pruned before the search of initial centers so that only those points, for which the number of available values exceeds 50% of the overall dimension of the data space, will remain as candidates for the initial centers. The rest of the data points are discarded, since they contain a quite small amount of information about the data. Moreover, the initial centers that contain missing values (i.e. sort of outliers) in more than half of the variables are expected to disturb the overall performance of any clustering method. In summary the main steps of the ModKKZ are:

1. Choose the data points with more than 50% of data values available. If the number of prototypes is greater than the number of data points containing at least 50% of the data values, then terminate the algorithm.

2. Find the data point that has the largest value with respect to the general dissimilarity measure (6) and assign it as the first initial center. If the data point contains missing values, fill them in by the Nearest-neighbor imputation algorithm 7.2.1.

3. Find the rest of the initial centers $k = (2, \ldots, K)$ as with KKZ (Algorithm 7.1.1) by applying the general dissimilarity measure (6) and imputation as in Step 1.

7.2.4 TrobBF and TModKKZ - robust initialization methods with trimming

In order to reduce the sensitivity of the algorithms to the outlying values, a trimming procedure can be applied to the data before running the robBF or ModKKZ initializations methods. Trimming can be justified by the assumptions arisen from the results by García-Escudero and Gordaliza [11]. If the sensitivity towards outlying data points can be reduced, the risk of empty or singleton clusters should be reduced as well.

Figures 33 and 34 present sample situations about the undesired effect of outliers when two clusters are searched from the bivariate small data set by using a robust clustering algorithm with KKZ initialization. In Figure 33, the K-spatialmedians clustering algorithm is initialized by the ordinary KKZ method. The initial points are represented in the left plot. They are marked by circles. The points are the most distant points with respect to each other, but one of them is clearly an outlier. In the right plot the subsequent cluster centers obtained by

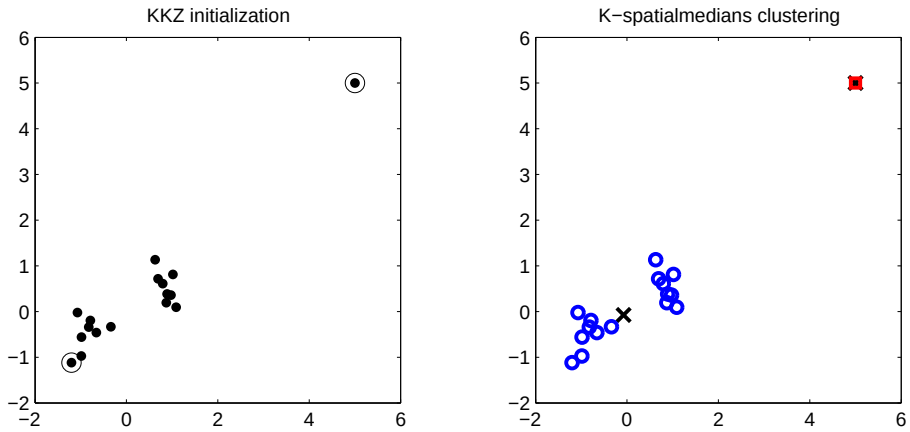


FIGURE 33 The problem with KKZ initialization. One outlying point prevents the algorithm from finding the useful clusters from the data.

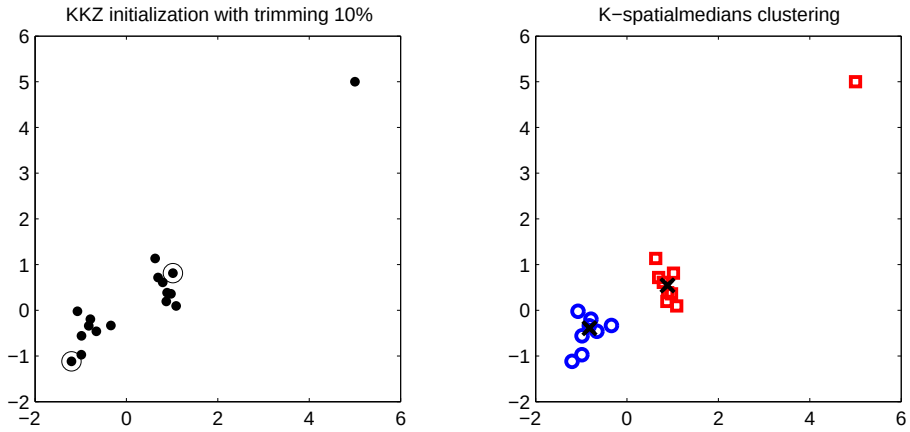


FIGURE 34 The robust behavior of the trimmed KKZ initialization and K-spatialmedians clustering.

the robust clustering algorithm are marked by '×'. A singleton cluster is created. More interestingly, the result seems to be the global minimum of the K-spatialmedians cost function. This verifies the results by García-Escudero and Gordaliza [11] that the robust clustering methods do not automatically inherit the properties of the robust estimates. In Figure 34, the trimming approach is added to the KKZ method. The fraction of trimmed data is 10%. Again, on the left, initial cluster centers are marked by circles. On the right, one can see that the robust clustering algorithm, K-spatialmedians, is no more biased by the individual outlier. The data contains one outlier, and in Figure 33 there are two clusters, but no outliers. In Figure 34, there are two clusters and one outlier, which means that the properties of the data are better maintained in the clustering.

The trimming is performed by computing the spatial median of the subdataset and using it as a reference point when evaluating the remoteness of the data points. The spatial median is considered a better estimate for the center of the main bulk of the data than, for example, the extremely sensitive sample mean. Moreover the recognition of the true outliers is thus easier. If, however, the case is that the data set does not contain any outliers, it follows that the trimming will re-

move some relevant objects and, thereby, some of the useful information. Hence, it is important to be careful with trimming and not to trim too large part of the data. For ensuring the use of all available information at least in some phase of the clustering process, trimming is applied only in the initialization phase and finalizing clustering is performed on the whole data. By this way, empty or very small clusters on the edge of the data cloud may be avoided. Moreover, the information about the outlying points is not totally lost, because even though they are not used during the initialization, they will be captured again by the closest clusters when the complete data is clustered.

The amount of the trimmed data is the most difficult issue from the DM point of view. It might be possible to use some heuristic for estimating the existence of the outlying points. For instance, by comparing the locations of the sample mean and spatial median of the full data set some information about the existence of outlying values could be achieved.

The use of the TrobBF method is otherwise similar to robBF, but the sub-datasets will be trimmed. The method is given in Algorithm 7.2.4.

Algorithm 7.2.4. TrobBF

Input parameters: Data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^p$ for all $i = 1, \ldots, n$, the number of clusters K, the number of sub-datasets J, the size of the sub-datasets $\tilde{n}$, and the trimming percent *trim*.

Output: K non-empty cluster centers $\{\mathbf{m}_k^*\}_{k=1}^K$.

Step 1. Set $\mathbf{X}_{ref} = \phi$.

Step 2. For $i = 1, \ldots, J$

Step 2.1. Draw $\tilde{n}$ data points with equal probability from X. Let us denote the resulting sub-dataset by $\tilde{\mathbf{X}}$.

Step 2.2. Compute the spatial median $\tilde{\boldsymbol{\mu}}$ of $\tilde{\mathbf{X}}$.

Step 2.3. Order the points in $\tilde{\mathbf{X}}$ according to the distance from the spatial median $\tilde{\boldsymbol{\mu}}$

Step 2.4. Remove *trim* percent of the most distant points in $\tilde{\mathbf{X}}$. Denote the trimmed sample by $\tilde{\mathbf{X}}_t$

Step 2.5. Find K cluster centers for $\tilde{\mathbf{X}}_t$ using Algorithm 7.2.2. Let us denote the centers by $\hat{\mathbf{C}}_i$.

Step 2.6. Extend the refined dataset as $\mathbf{X}_{ref} = \mathbf{X}_{ref} \cup \hat{\mathbf{C}}_i$.

Step 3. Set $\hat{\mathbf{C}}^r = \phi$.

Step 4. For $i = 1, \ldots, J$

Step 4.1. Take $\hat{\mathbf{C}}_i$ as the initial points and find K centers for $\mathbf{X}_{ref}$ by using the K-spatialmedians algorithm. Let us denote the set of center points by $\hat{\mathbf{C}}_i^*$.

Step 5. The refined centers are $\mathbf{C}^* = \arg\min_{\hat{\mathbf{C}}_i^*} Distortion(\hat{\mathbf{C}}_i^*, \mathbf{X}_{ref})$.

TModKKZ follows the ideas of KKZ and ModKKZ algorithms except that a given proportion of the data is removed with respect to the location of the spatial median. Hence, the main steps are the computation of the spatial median on the data, ordering the points into descending order with respect to the distance to the spatial median point, trimming of the most distant points and, finally, performing the ModKKZ algorithm for trimmed data. The detailed algorithm is not repeated here.

7.3 Numerical experiments on simulated data

Let $\boldsymbol{\mu}_k \in \mathbb{R}^p$ $(k = 1, \ldots, K)$ be the K true centers of the generating cluster distributions and $\mathbf{m}_k \in \mathbb{R}^p$ $(k = 1, \ldots, K)$ the K centers that are estimated by the chosen clustering algorithms (K-spatialmedians and K-means) over the full data set. Let $\pi(K)$ denote any permutation on $\{1, ..., K\}$. The error of the obtained clustering solution is defined by the permutation $\pi(K)$, which minimizes the distance between the true centers $\boldsymbol{\mu}_k$ that were used for cluster generation and the centers computed using the clustering algorithms over the full data set from a given initial starting point:

$$err = \min_{\pi(K)} \sum_{k=1}^{K} \|\boldsymbol{\mu}_k - \mathbf{m}_{\pi(K)}\|_2^2. \tag{58}$$

The error estimate for a particular method is the average error of 100 trials given by

$$\widehat{err}(\{\mathbf{m}\}_{i=1}^N) = \frac{1}{100} err. \tag{59}$$

In order to compare errors for different number of dimensions and clusters, the error estimates are scaled by

$$\overline{err} = \frac{1}{Kp} \widehat{err}, \tag{60}$$

where K is the number of clusters and p the data dimension. The estimated CPU time expresses the time elapsed from the start of initialization to the accomplishment of the final clustering process over 100 trials.

Seven different combinations of initialization and clustering methods were tested. K-spatialmedians clustering was tested together with six different initialization methods. As a comparison, the original K-means-based approach by Bradley and Fayyad was used to compare the K-means clustering with K-spatialmedians clustering when the non-robust BF method was used in the initialization. Hence, the following combinations were used in the experiments:

- BF + K-spatialmedians

- TrobBF + K-spatialmedians
- robBF + K-spatialmedians
- Random + K-spatialmedians
- BF + K-means
- ModKKZ + K-spatialmedians
- TModKKZ + K-spatialmedians

The size of the sub-datasets used in BF, robBF, and TrobBF was 10% of the overall data set. In TrobBF 10% of the sub-datasets were trimmed away. The initial centers for clustering of the sub-datasets were chosen in random. The stopping rules for the K-spatialmedians and K-means methods were that no more changes occur in the cluster assignments or the maximum number of iterations 1000 is accomplished. In the sub-dataset clustering the maximum iteration count was set to 100. The KKZ-based methods do not generally require initial parameters, but TModKKZ needs the percent of eliminated data, which was chosen to be 10. In all experiments, all variables were scaled to the range $[0, 1]$ before the clustering process.

7.3.1 Test 1: Compact, well-separated and spherical Gaussian clusters

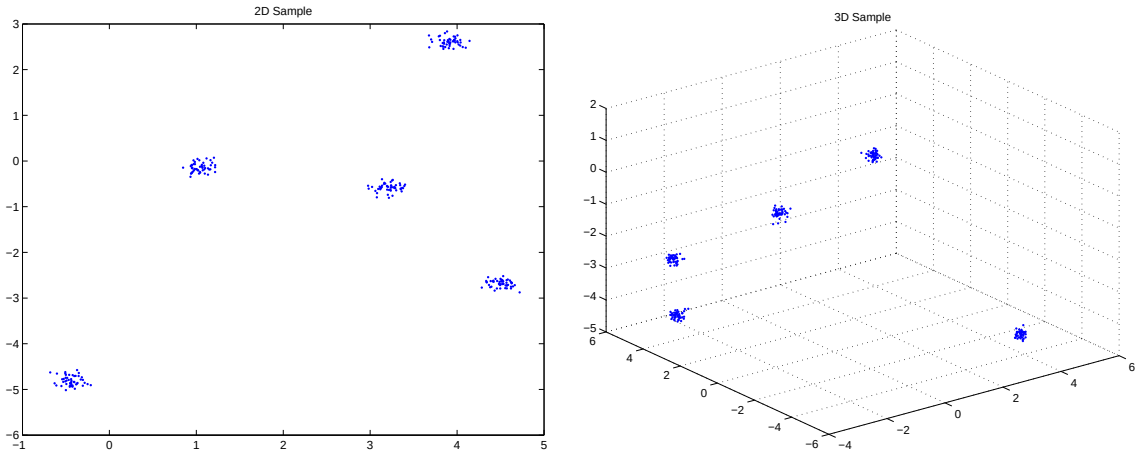


FIGURE 35 2- and 3-dimensional examples of the data sets used in test 1.

Clearly, it is practically impossible to test clustering methods with all different situations in mind. We first tested the methods on a set of very easy data sets. It is a minimum requirement for any clustering method to show accurate and efficient performance on well-separated clusters without noise and missing values. In order to evaluate this, we did a set of experiments on well separated 2-,5-,15- and 30-dimensional data sets with various number of clusters ($K = 2, 3, 4, 5, 6, 7, 8$). Two and three dimensional examples of the data sets are

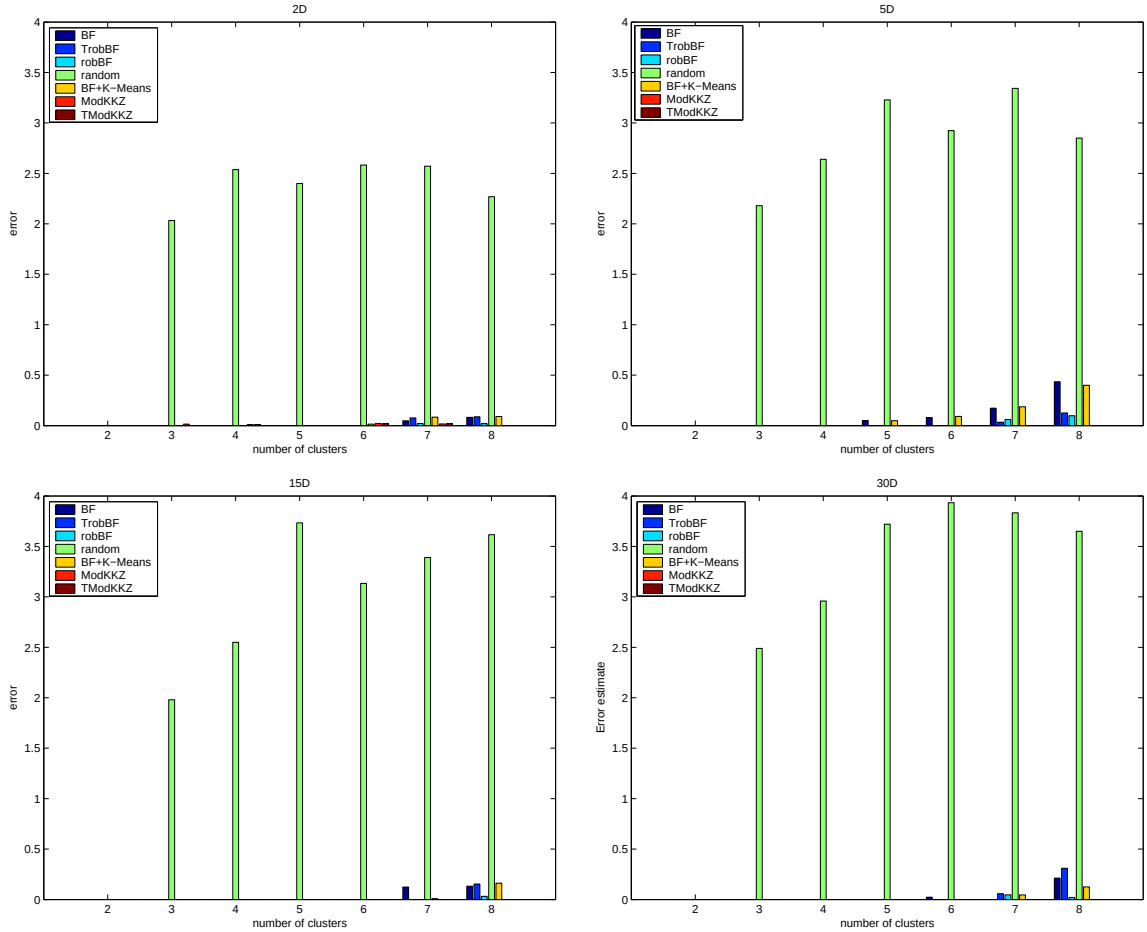


FIGURE 36 From left to right and top to down the mean estimates (N=100) of the scaled clustering error $\overline{err}$ on 2-,5-,15-, and 30-dimensional complete Gaussian test samples, respectively.

presented in Figure 35. The data sets were sampled randomly from a mixture of spherical and symmetric normal distributions defined by $\sum_{k=1}^{K} N_p(\mathbf{m}_k, 0.1 \times \mathrm{I}_p)$. All clusters consisted of 50 data points and were of equal size, which means that any feasible methods would not mistake the clusters for outliers. The components of center vectors $\{\mathbf{m}_k\}$ $(k = 1, \ldots, K)$ were restricted to range from -5 to 5 and the Euclidean distance between any pair of cluster centers was always greater than one. This means that each time when the generated cluster center $\mathbf{m}_{k+1}$ failed to satisfy the between-distance rule $\|\mathbf{m}_{k+1} - \mathbf{m}_{k'}\| \geq 1$ for all $k' = 1, \ldots, k$, it was discarded and regenerated. The relatively small within-cluster variance and the predefined minimum gap of the between-cluster distance makes the clusters well-separated. Noise or missing values were not added into the data sets. Using these settings, all relevant methods should perform relatively well. We computed the estimates for the squared errors, CPU times, and the number of clustering iterations needed to converge from the initial points generated by the different methods. The estimates were computed as the averages of one hundred tests runs.

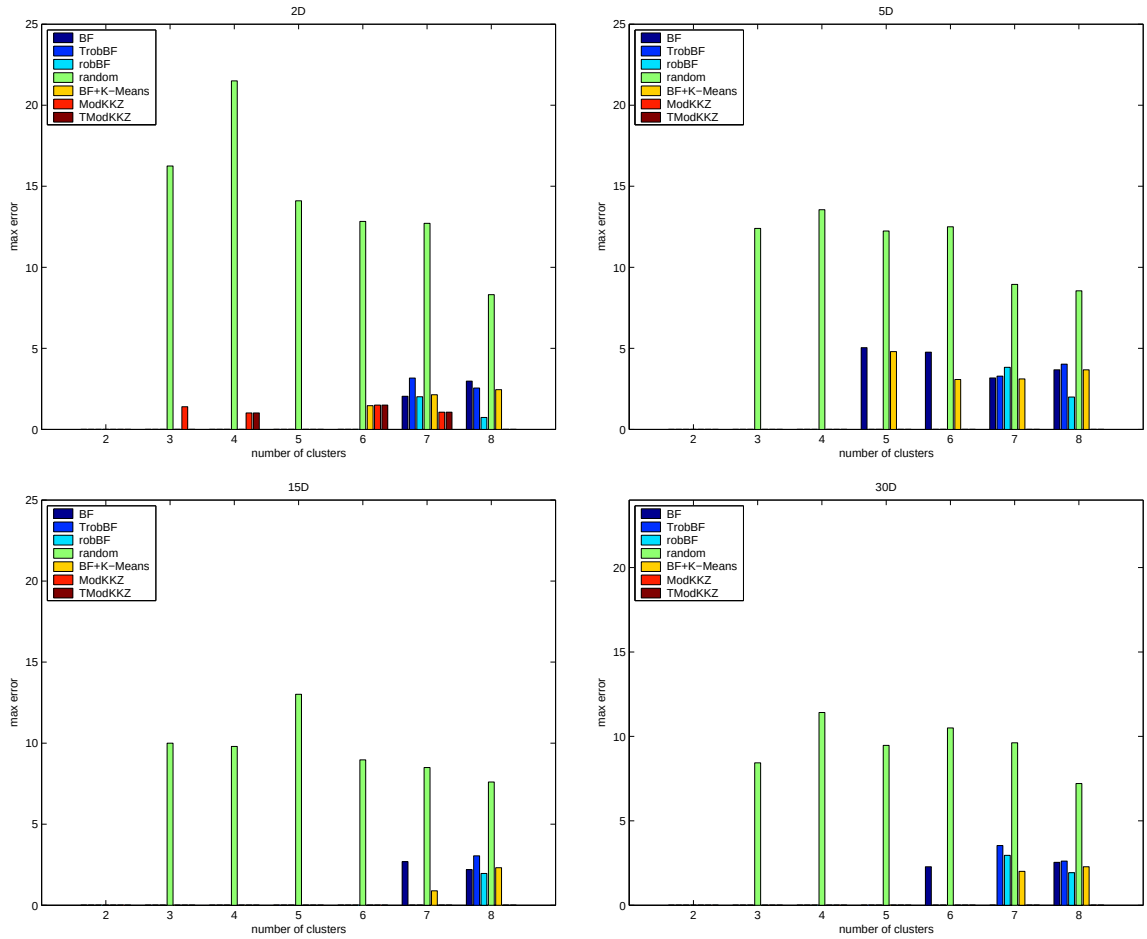


FIGURE 37 From left to right and top to down the maximum of the scaled clustering error $\overline{err}$ from 100 trials on 2-,5-,15-, and 30-dimensional complete Gaussian test samples, respectively.

Comparison of error estimates

As it is expected, in average all the methods produce quite good results except the random initialization (see Figures 36 and 37). It is viable for two cluster data, but for $K \geq 3$ the quality collapses. Plots in Figure 38 represent the average and median estimates of the scaled error $\overline{err}$ when the K-spatialmedians clustering was performed using random initialization. It seems that the error does not increase anymore when $K > 3$, but it increases along the dimensionality. The mean estimates for the other methods are acceptable. For $K \in \{5, 6, 7, 8\}$ slightly increased errors are shown especially for the non-robust BF-initialization and it does not matter whether the K-means or K-spatialmedians method is used as the final clustering method. One can also see from Figure 37 that even if the average error estimates for the BF-initialized methods are quite small when $K = 5$ or $K = 6$, some bad results already exist. Another problem with the BF-initializations are the empty clusters. Table 5 shows that the K-means-based initialization is much more sensitive to empty clusters than its robust variants. Even the random initialization performed better than the K-means-based meth-

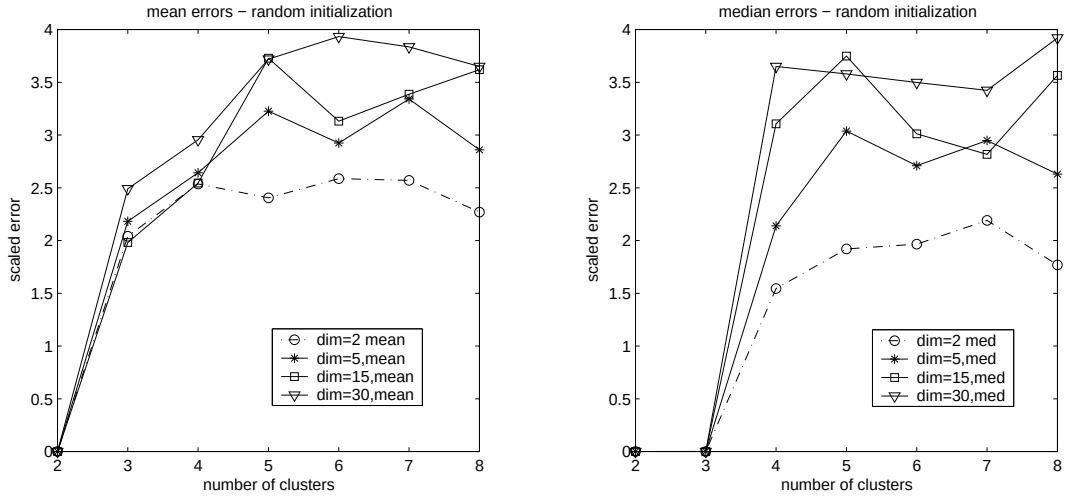


FIGURE 38 The mean and median estimates (N=100) of the scaled error $\overline{err}$ for K-spatialmedians clustering from random initial points.

TABLE 5 The number of trials that led to an empty cluster on noise- and error-free normally distributed test cases. The total number of trials in each case is 800.

p	BF	TrobBF	robBF	random	BF-K-means	ModKKZ	TModKKZ
2	6	1	1	0	10	0	0
5	36	2	1	1	41	0	0
15	15	0	1	6	13	0	0
30	13	2	0	4	8	0	0

ods in this sense. Hence, according to the results, it seems that the use of robust estimates in density-estimation based initialization is effective even if the clusters are sampled from the complete, well-separated, and error-free normal distributions. In some rare cases, the results of TrobBF are slightly more disturbed than the results from the non-trimmed procedure robBF. This is likely due to the fact that, in error-free cases, the density-based trimmed initialization methods are more dependent than the non-trimmed variants on the size of data. This is obvious since on non-erroneous data correct sample points are removed in trimming, which leads to inefficiency in statistical estimation. It is interesting to note that such a problem is avoided by using the trimmed distance-optimization method TModKKZ. TModKKZ searches mutually distant points and if any is left in a distant cluster after trimming, the cluster will be taken into account in the initialization. Hence, the statistical efficiency is not as much dependent on the number of members in the cluster. Notice that this property of TModKKZ could be further exploited by initializing the sub-clustering processes of the density-based initialization methods by distance-based methods.

Comparison of computational costs

CPU times are presented for different numbers of clusters and dimensions in Figures 39 and 40, respectively. The clearly fastest solutions are obtained with the

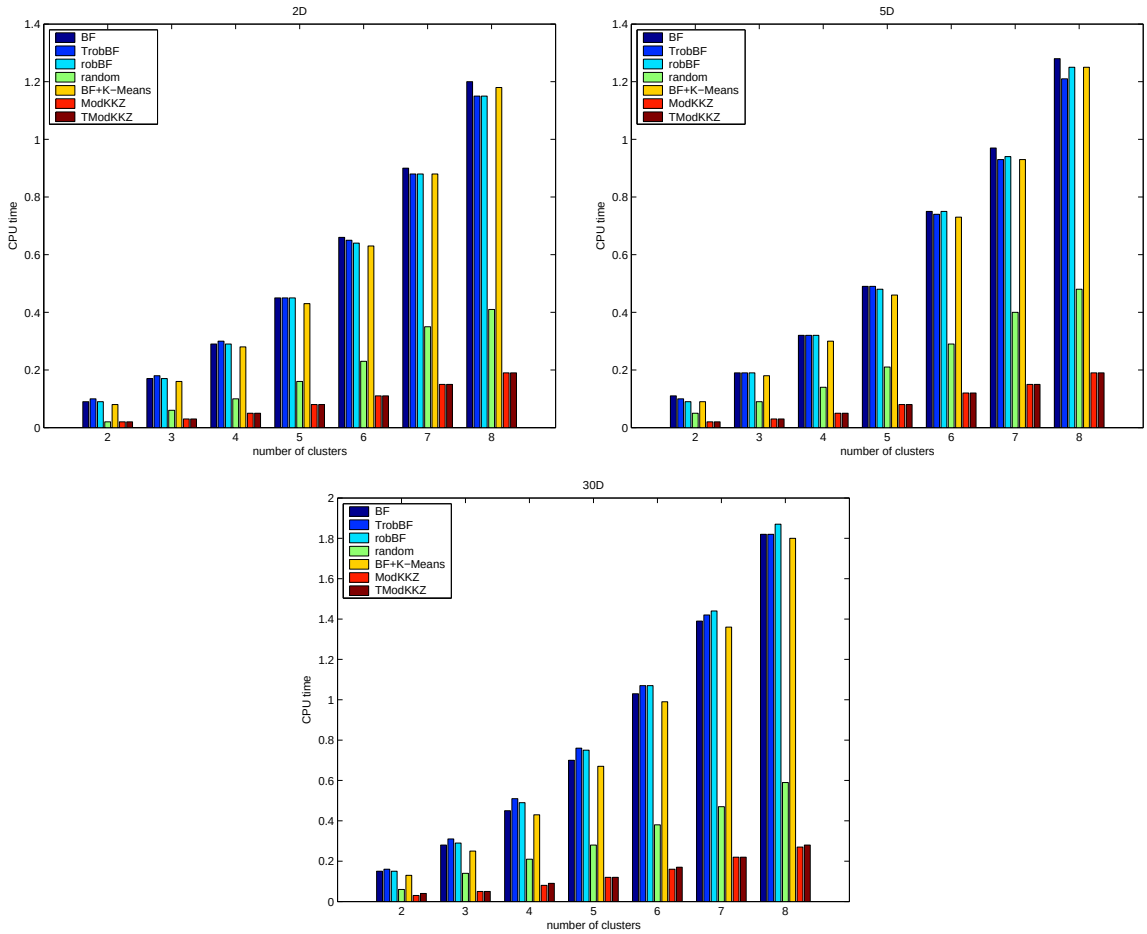


FIGURE 39 CPU times on 2-,5-, and 30-dimensional clean data.

distance-optimization methods ModKKZ and TModKKZ. The estimated CPU time of the K-spatialmedians clustering combined with the ModKKZ or TModKKZ initialization is even smaller than in random initialization. This is due to the smaller number of clustering iterations taken from the initial points (see, Figure 41). The required computation time by BF, robBF, TrobBF and BF+K-means are almost equal to each other. Although one iteration taken by the K-means algorithm is faster when compared to the K-spatialmedians algorithm, the latter usually needs fewer iterations to converge. From this it follows that the overall time required for the clustering process is shorter for the robust K-spatialmedians clustering.

Discussion

In summary, the distance-optimization initialization gives better quality for the final clustering methods than the density-estimation or random initialization even on these relatively well-clustered data sets. This holds with respect to the size of errors in the final clusterings, the number of empty clusters, and CPU time. The results show that the random initialization does not perform well even on these well-separated cluster samples. In addition to the quality of the clustering results,

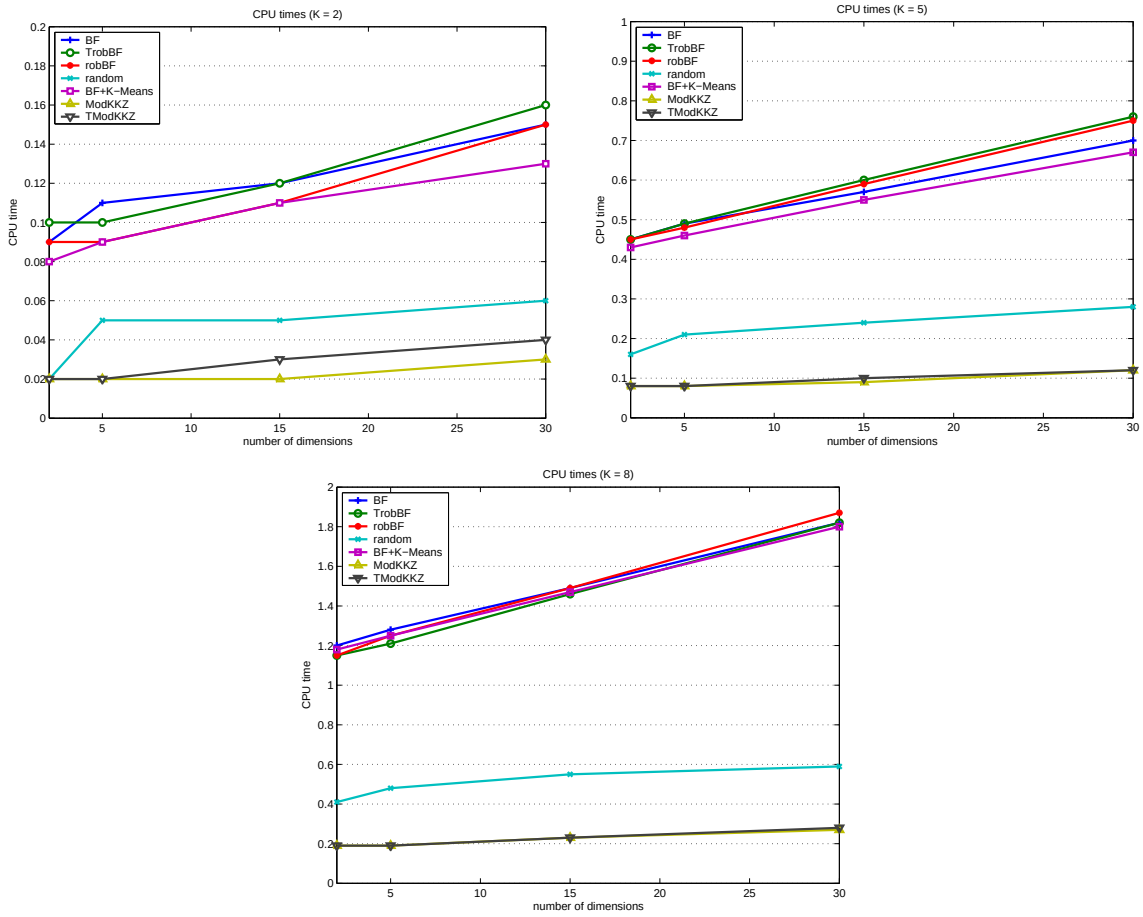


FIGURE 40 CPU times on 2-,5-, and -8 clusters data sets.

the overall time to converge is poor for the random initialization as well. This is due to poorly chosen initial points, starting from which it takes more iterations by final clustering algorithm to converge.

When the density-estimation initialization is applied, the robust methods seem to overtake the K-means-based methods. They produce smaller errors and are clearly less prone to empty clusters. However, the trimming leads to larger errors with robust variants as well. These results favor distance-optimization initialization methods. The following experiments will evaluate the effects of more complex situations, such as missing values and outliers, to the methods.

7.3.2 Test 2: Compact, well-separated and spherical Gaussian clusters with missing data

The experiments were performed using the same test configurations as in Test 1, but missing values were uniformly generated into the data. The obtained errors are presented in Figure 42. When compared to the complete data cases, one can easily note that the errors are significantly larger when missing data is present. The only exception is observed in the two clusters case, in which all methods produced almost the same average error as in the complete data cases. The dis-

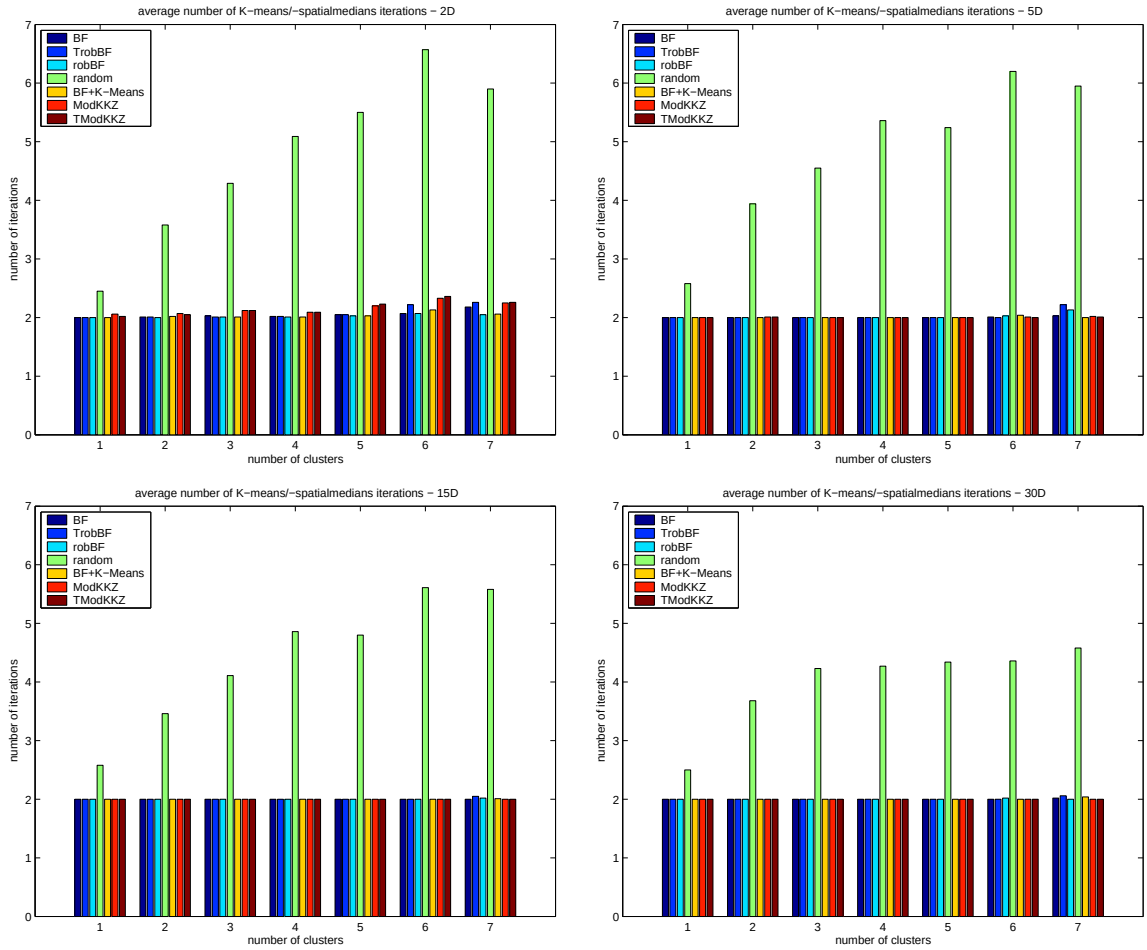


FIGURE 41 The number of iterations taken by the clustering methods after different initialization methods.

tributions of the errors, CPU times, and final clustering iterations are illustrated by histograms in Figures 45 and 46. The means and medians of the results are depicted by vertical lines and denoted by μ_e, μ_t, and μ_i for the means, and med_e, med_t, and med_i for the medians, in the figures. From the histograms in Figure 46 one can see that the large average error estimate of TrobBF on the two cluster test with 45% of missing data is caused by a single significantly erroneous test run and the median estimate for the error equals that in the other methods. A distinguishable average error estimate was also obtained with the random initialization on the two cluster case when 15% of data is missing. Also in this case the median estimate does not differ, since the error is caused by only one individual gross error during the test runs (see Figure 45).

As in the case of complete data, the worst error estimates on the incomplete data sets are obtained by the random initialization method. The K-means-based initialization methods also ended up with quite poor quality solutions in the presence of missing data.

ModKKZ is clearly the most efficient initialization method when comparing the quality of the estimates in the final clustering solutions. Whereas ModKKZ is

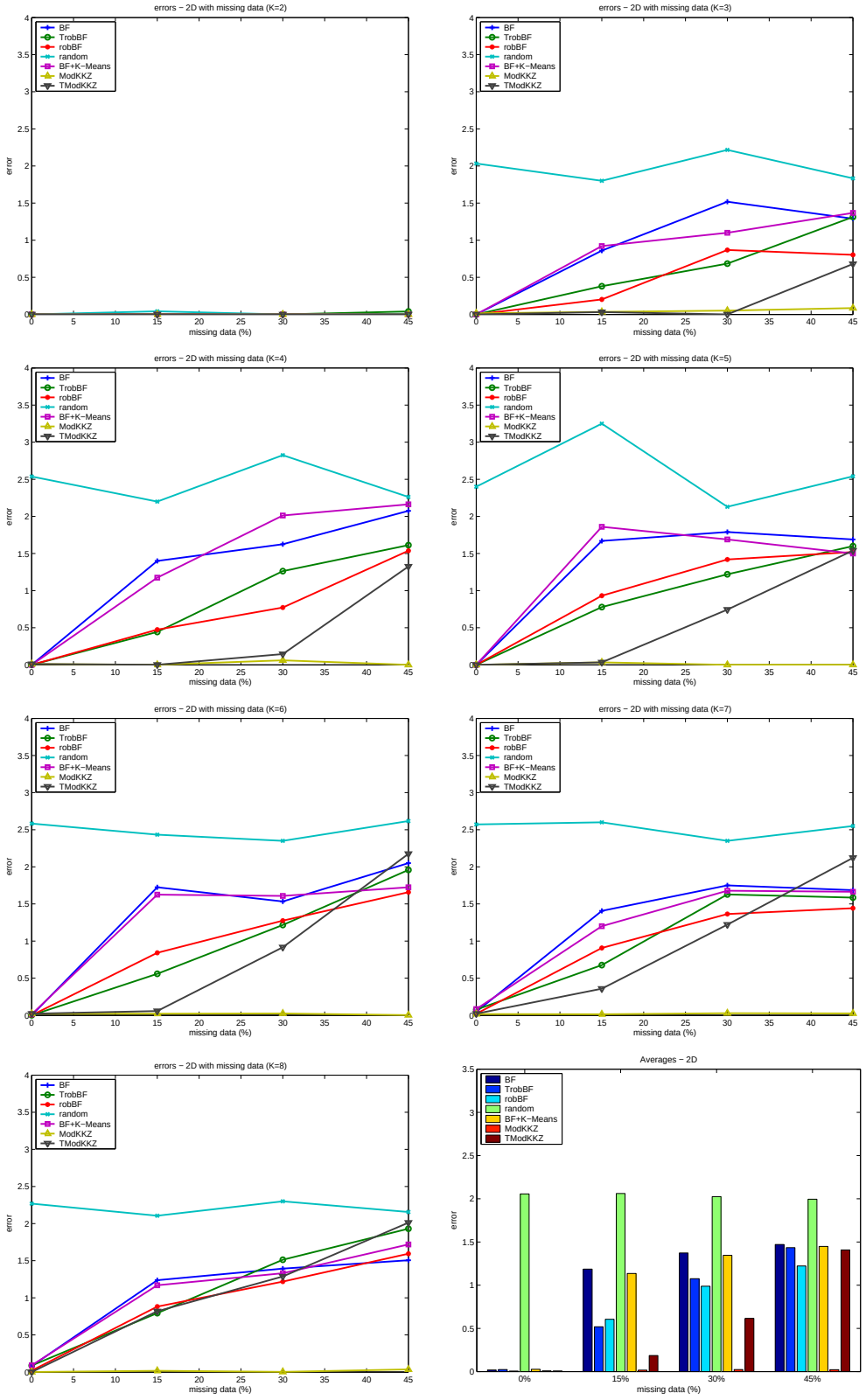


FIGURE 42 The mean estimates (N=100) of the scaled clustering error $\overline{err}$ on 2-dimensional incomplete Gaussian test samples with different number of cluster and missing data values.

TABLE 6 The numbers of empty clusters with respect to missing data on the 2-dimensional data sets.

%	BF	TrobBF	robBF	random	BF+K-means	ModKKZ	TModKKZ
0	6	1	1	0	10	0	0
15	2	1	1	0	11	0	0
30	2	0	0	0	2	0	0
45	3	0	0	0	2	0	0

the best one also in this case, the trimmed variant TModKKZ suffers quite badly from the missing data values. The average quality of the TModKKZ solutions is even lower than the K-means-based sub-sampling-methods when 45% of the data is missing and there are more than four clusters. For the same settings, ModKKZ performs extremely well. One should note that when 30% or less of data is missing, the estimated error by TModKKZ is the second lowest. Removing 10% of data points, that is the applied trimming fraction, simultaneously with 45% of missing data values leads to a loss of more than half of the data during the initialization. If, at the same time, the number of clusters is high, some clusters may almost disappear from the data. Such a loss of information may lead to serious errors with the distance-based methods. Interestingly the same behavior is not observed with the trimmed sub-sampling based method. This might be due to the multiple subsampling, which assures the more thorough use of all data. Hence, TrobBF does not suffer from the trimming as much as TModKKZ. This may indicate that the sub-sampling-based methods are more robust than distance-based methods as the fraction of missing data approach to 50%. It seems also that there are no big differences between the robust and K-means-based sub-sampling methods when 30% or 45% of data is missing. This is not supposed to be the case if also some outliers were present in the data. It seems that the amount of missing data does not influence remarkably the sensitivity to empty clusters. Table 6 shows that worst performers in this sense are BF and BF-K-means as they were in the case of complete data. The numbers show that larger fractions of missing data enhance the performance of the methods rather than impair it. Other than K-means-based methods do not suffer from the problem of empty clusters at all.

It is also important to compare how much time it takes to operate on incomplete data by the different initialization methods. It is clear from the results in Figure 43 that the random and distance-based initializations have superior CPU times. The differences in the computation times of the sub-sampling-based methods are not noteworthy. The robust methods robBF and TrobBF converge slightly faster than BF and BF-K-means, which is due to smaller number of K-spatialmedians iterations accomplished during the final clustering. The numbers of iterations used by the K-means and K-spatialmedians algorithms for convergence from the different initializations are depicted in Figure 44. On the complete data sets, the number of the final clustering iterations seemed to be quite independent of the number of clusters. Only if the random initialization was used,

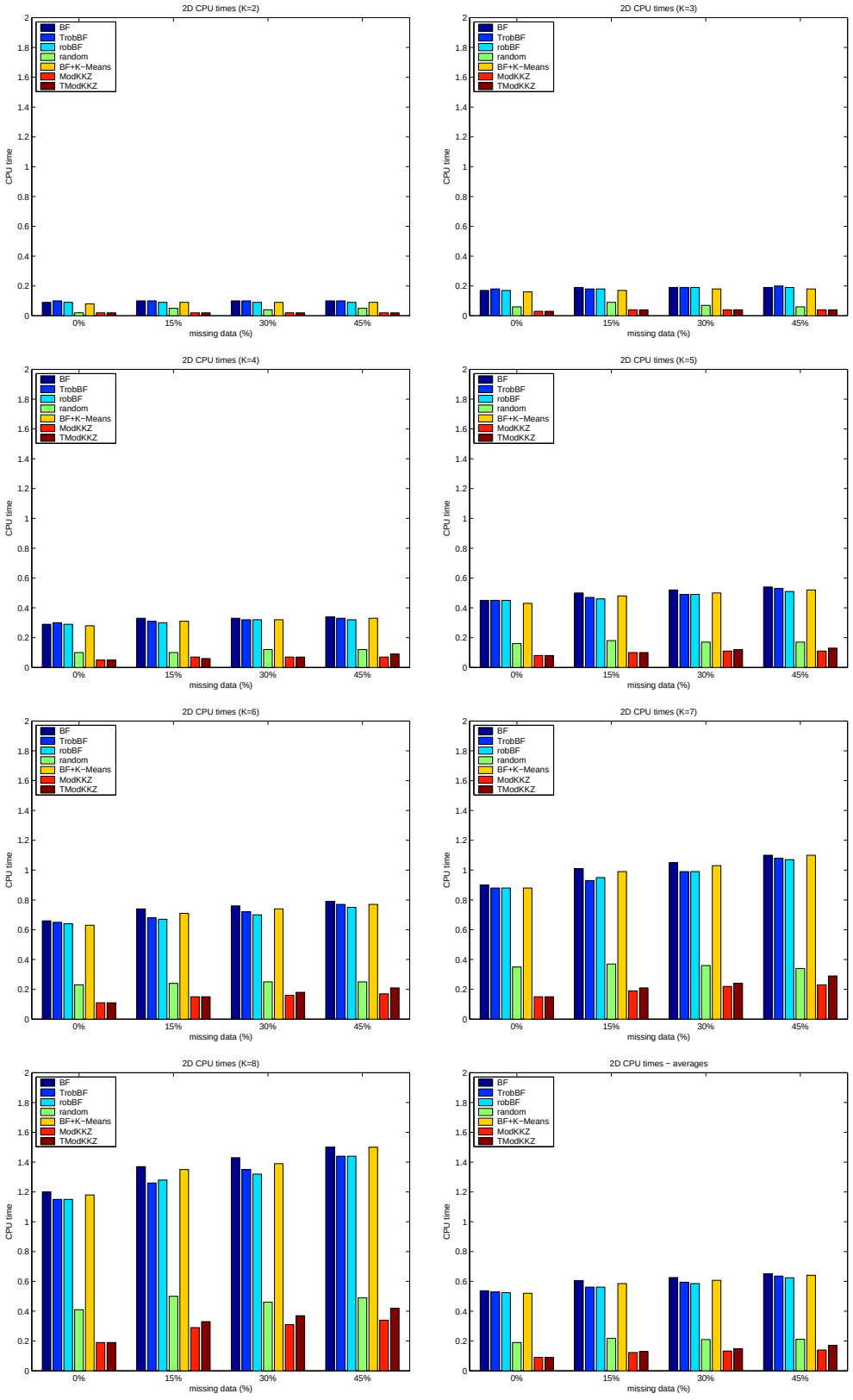


FIGURE 43 Average CPU times from the tests on the incomplete data sets. On the bottom-right there are the averages of the bars.

the number of iterations grew with respect to the number of clusters (cf. Figure 41). This is no more the case in the presence of missing data. The number of final clustering iterations grows both along the number of clusters and amount of missing data. In most cases robBF, TrobBF, and ModKKZ require the smallest number of clustering iterations. The only method that sticks out in the tests is the random initialization that needs a large number of clustering iterations in all cases. The differences between the others are quite small. Perhaps the most remarkable detail of the graphs is the growing numbers of clustering iterations when the TModKKZ initialization is used. As the overall CPU time is very small for TModKKZ initialized clusterings (Figure 43), it means that the initialization process itself has to be very fast. One should also note that a higher number of K-spatialmedians clustering iterations does not automatically mean a good quality clustering solution (cf. Figures 42 and 44).

7.3.3 Test 3: Scalability in data size and dimensions

As the focus of this study is mainly to settle down to the KM/KDD/DM aspect, it is important to test whether the algorithms are efficient and scalable on large data sets or not. Thereby, bearing this on mind, tests were also performed on a couple of relatively large data sets. The influence of increasing the size of clusters (more data points in a cluster) and dimensions were investigated.

Figure 47 provides some graphical information about scalability of the algorithms with respect to the number of data points. The number of clusters and dimensions are fixed to $K = 3$ and $p = 3$, respectively. The results describe the average of the estimates over 100 simulation runs. The average CPU time required by the random initialization is clearly the highest. It is neither fast on small scale problems nor scalable to large clustering problems. The average error of the solutions is such that it takes far too many iterations, and thereby, far too much time, to obtain stabilized solutions on large DM data sets. Hence, although so many times being considered as the *de-facto* solution in the search for the globally best partition by the fast K-means algorithm, the multiple random repetition still seems to be far from optimal when one is dealing with large-scale data sets, as is usual in the DM context. The distance optimization methods, ModKKZ and TModKKZ, outperform the density estimation methods in the scalability issues. The better performance is due to the simplicity of the initialization methods themselves, because the initial configurations obtained by any of these methods lead to about equal number of k-clustering iterations. Unlike with the random initialization, the errors for the solutions are very small with all the other methods. The standardized errors are almost independent of the sizes of clusters. A couple of solutions with somewhat larger errors are obtained by ModKKZ initialized K-spatialmedians and BF initialized K-means clusterings. Nevertheless, the errors are quite insignificant.

While the previous results describe the behavior of the different initialization algorithms with respect to the growing amount of data in the clusters, the following results present the influence of dimensionality (See, Figure 48). Also in this case the scalability of the random initialization can not compete with the other methods, because it takes nearly three times more clustering iterations to converge from the randomly chosen prototypes. The overall CPU time grows clearly faster for the random initialized clustering than for the other approaches. The use of the K-means approach either in the initialization step only or in the initialization and the final clustering step, seems to decrease the CPU time of the density based methods, which are slightly faster than their robust variants. The distance-based methods also scale best in this case. The trimmed variants of the distance optimization and density estimation seem to require somewhat more computation with respect to the number of dimensions, but the differences are not remarkable in these tests. As the errors are approximately equal for all methods, except for the random method, the distance-optimization methods are the most efficient methods when the clustered data includes a large number of dimensions.

TABLE 7 Test parameters.

data	K	p	n_k	n	miss%/cl	noise%
A	3	30	$220-260$	720	33	30
B	6	30	$120-180$	900	33	10
C	3	50	$220-260$	720	0-40	5
D	7	50	$100-300$	1455	0-50	15

Because the simulated data sets used in these tests comprised of extremely clearly separated cluster structures, some experiments are needed to be performed under more difficult conditions, where the clusters are of arbitrary shape, and noise, and outliers and missing data exist.

7.3.4 Test 4: Clusters of arbitrary shapes with noise and missing data

In order to evaluate the performance of the different initialization methods under more disturbed conditions, experimental settings that produce more noisy and incomplete clustered data sets were defined. Table 7 presents the parameters used. The total number of data points, n varied from 720 to 1455. The relatively high-dimensional data sets contained 3-7 clusters. The size of the clusters, n_{cl}, and in the case of C and D the amount of missing values per cluster as well, varied between the clusters. Moreover, 5-30% of the data were randomly transformed into uniform noise in the min-max range of the data. For keeping the proportion of missing data unchanged, empty values were not allowed to be filled with noisy ones. The values of the cluster centers were sampled uniformly from the interval $[-5, 5]$ for each variable. The minimum Euclidean distance for two cluster centers was 1.5 and the standard deviations of the Gaussian cluster centers and the scatter parameters of the Laplace clusters varied uniformly from 0.3 to 0.9. Hence, the clusters were not necessarily of spherical shape. The error, CPU time, and the clustering iteration estimates are computed from 100 iterations. Figure 49 presents two low-dimensional examples of the data sets used in the experiments.

Interpretation of the results

The results for data sets A,B,C,D are given in Figures 50-53. Different assumptions underlying the methods become clearly visible in these results and the success of each method depends heavily on the distributional conditions used in the data generation. The random initialization is also much more competitive when compared to the previous well-separated cases. The interesting thing is also that empty clusters were not met on these data sets. Let us first concentrate on the sub-sampling initialized variants and on the effects of robustness. The non-robust BF+K-means combination produces clearly the worst results of the sub-sampling-based initialization methods. It produces the largest estimates for the clustering error on all data sets except one. The exception is encountered in data B, in which

the BF initialization with the K-spatialmedians clustering gives the largest errors. Hence, robustness seems to provide an advantage on these noisy and incomplete data sets. TrobBF produces the smallest clustering errors on data sets B and C. Moreover, robBF is as good as TrobBF on data C. When the non-robust BF initialization is used, K-spatialmedians clustering gives better results than K-means on three of the four data sets. On data B, K-means with BF produces smaller average error estimates than K-spatialmedians, but the behavior is anyway very similar. The difference is only due to a couple of unsuccessful runs that have produced very large errors for the latter (see distributions in Figure 51). Hence, these results also confirm the general assumptions behind this study, i.e., that robust clustering methods are more likely to provide better clustering solutions under disturbed and incomplete conditions. They are not as easily misled by errors and missing values as the methods based on sample mean estimators.

The results do not give a straight answer about the utility of the trimming operation used in the robust TrobBF method. On B and C data sets the results are very good for TrobBF. However, on three of the four data sets the results by the non-trimmed robBF procedure are as good as, or even better than with TrobBF. Although it can be expected that the larger the amount of noise, the better the behavior of TrobBF when compared to robBF, this is not observed in these tests, since B and C data sets contain the smallest numbers of noisy values. Neither does the number of clusters nor the amount of missing values explain the differences. Hence, on this basis it is not possible to give general and precise recommendations about which one to choose for a clustering task.

As the general distributional assumptions are disturbed, the random initialization seems to be more and more efficient. It produces the smallest error estimates on two data sets (A and D) that are actually the most noisy ones (30% and 15%). Data set D can be considered a very difficult case, because there are seven clusters, the number of data points varies from 100 to 300, the cluster-wise fractions of missing values are 0%, 20%, 40%, 15%, 35%, 50% and 5% and, moreover, 15% of noise exist in the data. Furthermore, as the cluster data are generated from Gaussian and Laplace distributions with equal probabilities, it is extremely difficult to define the general assumptions for such data. Hence, the fact seems to be that the more unstable the conditions, the more biased the initializations of the heuristic methods become, because they do not assume noisy, arbitrary shape clusters with cluster-wise varying amounts of missing data.

The promising performance of the distance-optimization methods on well-separated cases presented in the previous sections fell down on the tests on unsteady data. ModKKZ gives the second and third smallest errors on data A and B, respectively, but produce totally unusable solutions for data C and D. The trimmed variant, TModKKZ, is also acceptable for data A, but produces many large errors for data B and totally unusable results for data C and D. The fractions of missing data varies cluster-wise in data C and D. As this is the most significant difference when the test parameters of A and B are compared to the C and D, it may be also the problem for the distance-based methods as the definition for the general distance between the two data points with unequal number of missing

values is hard to give.

In addition to the large clustering errors produced by ModKKZ and TModKKZ, their CPU times have also grown when compared to the results given earlier. This is due to the increased number of the finalizing K-clustering iterations. For example, the average numbers of K-spatialmedians iterations on case B are 10.00 and 11.61 whereas the clustering algorithms take approximately four iterations to converge after the density-estimation-based initializations on the same case (see Figure 51). Hence, the gap in the overall computation time is not that large after all. Despite the relatively large numbers of K-spatialmedians iterations, the random initialization has the shortest CPU times in cases A and B. TModKKZ has the shortest CPU times in C and D, but the random initialization is also very competitive in these cases. The density-based initialization is computationally a more intensive approach than the random or distance-based initialization, but it leads to fewer K-clustering iterations. This is an important property on large data sets, as the clustering iterations over the whole data set are expensive. For instance, in case D (see Figure 53), the overall CPU times of the density-estimation initialized methods are very close to the times needed by the KKZ-type or random methods. This points out the increased computational cost caused by clustering iterations on full data, which is an important detail from the DM point of view.

7.4 Conclusions

The numerical experiments under the previously described simulated conditions have shown the difficulty of finding a general initialization method and an overall clustering approach that would produce unique clustering results on all kinds of erroneous and incomplete data sets. In the DM/KDD context this confirms that the domain expertise is worth a great deal. The obtained results show that under nearly ideal conditions, which here refers to data sets that are composed of well-separated clusters, the random initialization approach and the sensitive K-means-based initialization methods are outperformed by robust density-estimation-based or distance-optimization-based initialization methods. The proposed modified initialization methods do not only give smaller errors, but require less CPU time as well. This is mainly due to the reduced number of the final clustering iterations. This is a very important result from the DM perspective as the cost of one full-data clustering iteration is higher on large data sets.

Tables 8,9, and 10 show the ranks of the different methods on complex data sets A,B,C, and D. Based on the results, some advice to the initialization problem can be given. On error-free complete data sets, the distance-optimization methods yield clustering solutions with the best quality in the shortest time. As Figures 33 and 34 illustrate, in the presence of outliers, it may be useful to trim the data before the initialization. This prevents empty or very small outlying clusters covering the true properties of the data as presented in Section 7.2.4. The exper-

TABLE 8 The rank of the methods according to the mean errors μ_e taken from Figures 50-53.

data	BF	TrobBF	robBF	random	BF+K-means	ModKKZ	TModKKZ
A	5	7	4	1	6	2	3
B	7	1	4	2	6	3	5
C	1	1	1	5	4	6	7
D	3	4	2	1	5	6	7

TABLE 9 The rank of the methods according to the means of the total CPU time μ_t taken from Figures 50-53.

data	BF	TrobBF	robBF	random	BF+K-means	ModKKZ	TModKKZ
A	5	7	6	1	4	1	3
B	5	6	6	1	4	2	3
C	5	7	6	2	4	2	1
D	5	7	6	2	3	4	1

TABLE 10 The rank of the methods according to the means of the K-means/K-spatialmedians iterations μ_i taken from Figures 50-53.

data	BF	TrobBF	robBF	random	BF+K-means	ModKKZ	TModKKZ
A	3	2	1	4	6	7	5
B	4	2	1	5	3	6	7
C	3	2	1	7	4	6	5
D	2	3	1	6	4	7	5

iments on the incomplete data sets show that ModKKZ clearly outperforms the rest of the methods even in the presence of 45% missing data values, but TModKKZ suffers from the missing data, and the quality of its solutions approaches the density-estimation methods as the fraction of missing values increases. The amount of uniform missing data, by itself, does not seem to have dramatic effect to computational costs of the methods. The best scalability with respect to the data size is obtained by the distance-optimization methods when the clusters are clearly separated.

Although being clearly the best performers on well-clustered data, ModKKZ and TModKKZ do not work well on more real-world-like data. When the distributional conditions behind the data become more complex, the robust variants of the density-estimation initialization methods outperform them in finding the cluster centers. The random initialization becomes also more competitive when the data is more dirty, in other words, erroneous and incomplete. As the quality of the clustering solutions obtained by ModKKZ and TModKKZ methods collapse on the high-dimensional and relatively large messy data sets, they can not be considered as the generally best choices for the DM clustering tasks that often focus on such data sets. When the distance-optimization methods are used, one should be satisfied that the data contains relatively well-separated clusters without noise and non-uniform missing data. Otherwise, robust density-estimation methods and random initialization would be the most promising methods. This study is thus not able to deny the *de-facto* position of the random initialization, but it provides new, robust, and fast methods for the initialization problems for the DM clustering tasks.

One of the main contributions of this chapter is the extensive tests performed for the number of existing and modified methods. The results have shown that the use of the robust elements in any part of the clustering process leads to better results in very different conditions. As surmised earlier on, universal methods for clustering problems cannot be given. The best of these methods can be distinguished, however, by their number of good properties, such as:

- Robustness
- Scalability
- Missing data treatment
- Minimal number of input parameters (input: data+K, output: clusters)

7.4.1 Future ideas

As already mentioned in the course of this chapter, the obtained results offer many novel possibilities for the further development of the automatic clustering algorithms. The methods based on the refined data sets by clustering subdatasets produce simultaneous information about variability in the cluster assignments and locations. This information might be useful to exploit in the stability- and prediction-based methods for estimating the number of clusters (cf., [336,

243, 291, 370, 95]). Furthermore, the density and distance based initialization methods could be combined into "hybrid methods" that might give better and more universal performance. An interesting issue is also the distance computation in the presence of missing data. In trimming, the data could be arranged with respect to the number of existing variables. This is based on the assumption that the more information about the similarity of two objects, or, in other words, the more common features available in two objects, the more closer to each other they are. A missing value does not contain information about the individual component of its data point, for example, be it an outlier or not. On the other hand, one should be careful with the treatment of missing data, since it may encompass a lot of information about the data set as a whole. Dorian Pyle [324] emphasizes that missing data patterns contain sometimes the most important piece of information for modelling. Therefore, it is important to take care that the information related to missing data is not completely lost during data processing. The amount of information in missing data as dependencies, correlations etc. depends on the missing data mechanisms that are discussed in more detail in Chapter 3. The development of these methods requires, nevertheless, a great number of experiments on real-world data sets and are is therefore left outside this work as a future issue to be addressed.

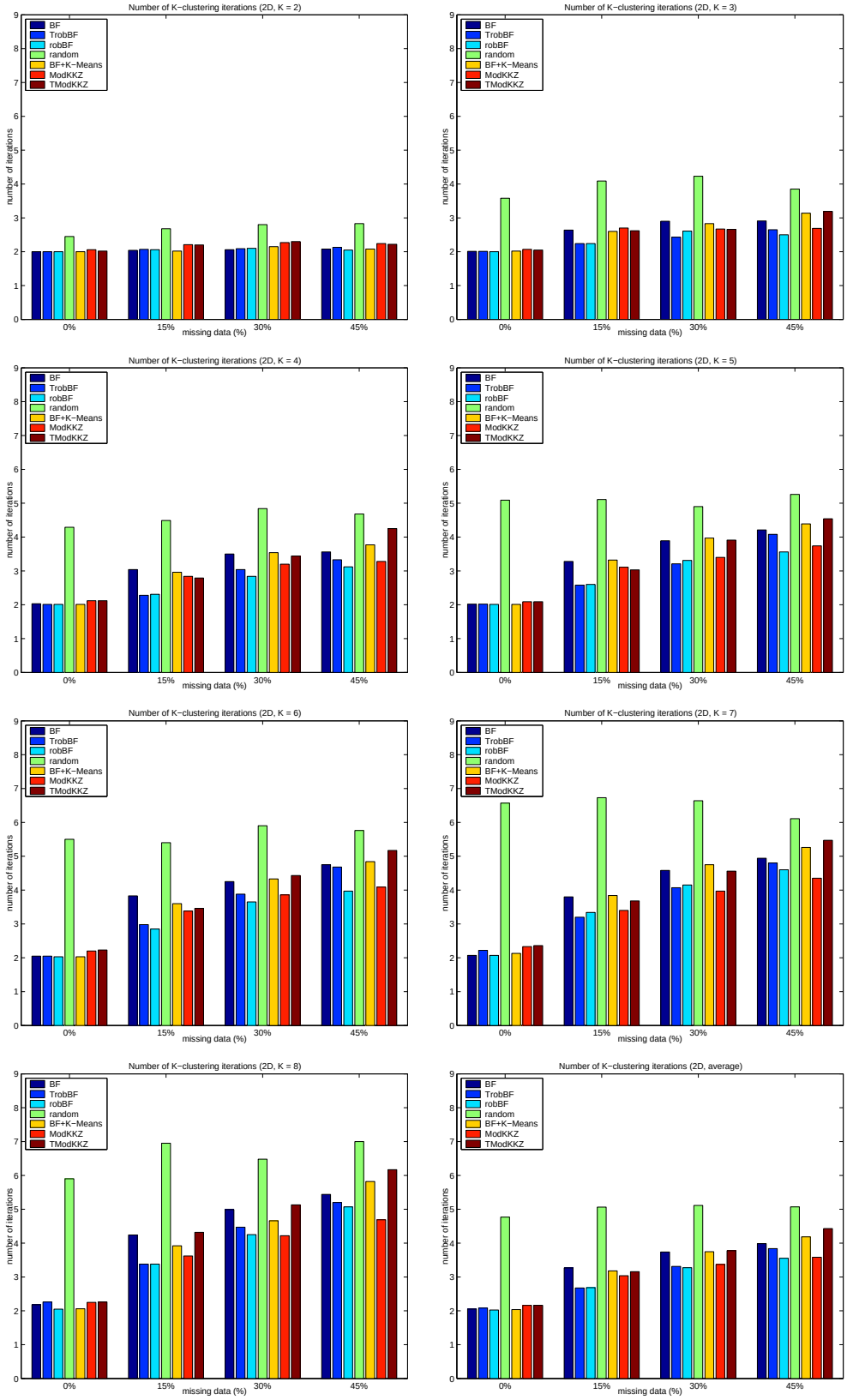


FIGURE 44 The average numbers of clustering iterations taken from the initial points on the incomplete data sets. On the bottom-right, the averages of the bars are shown.

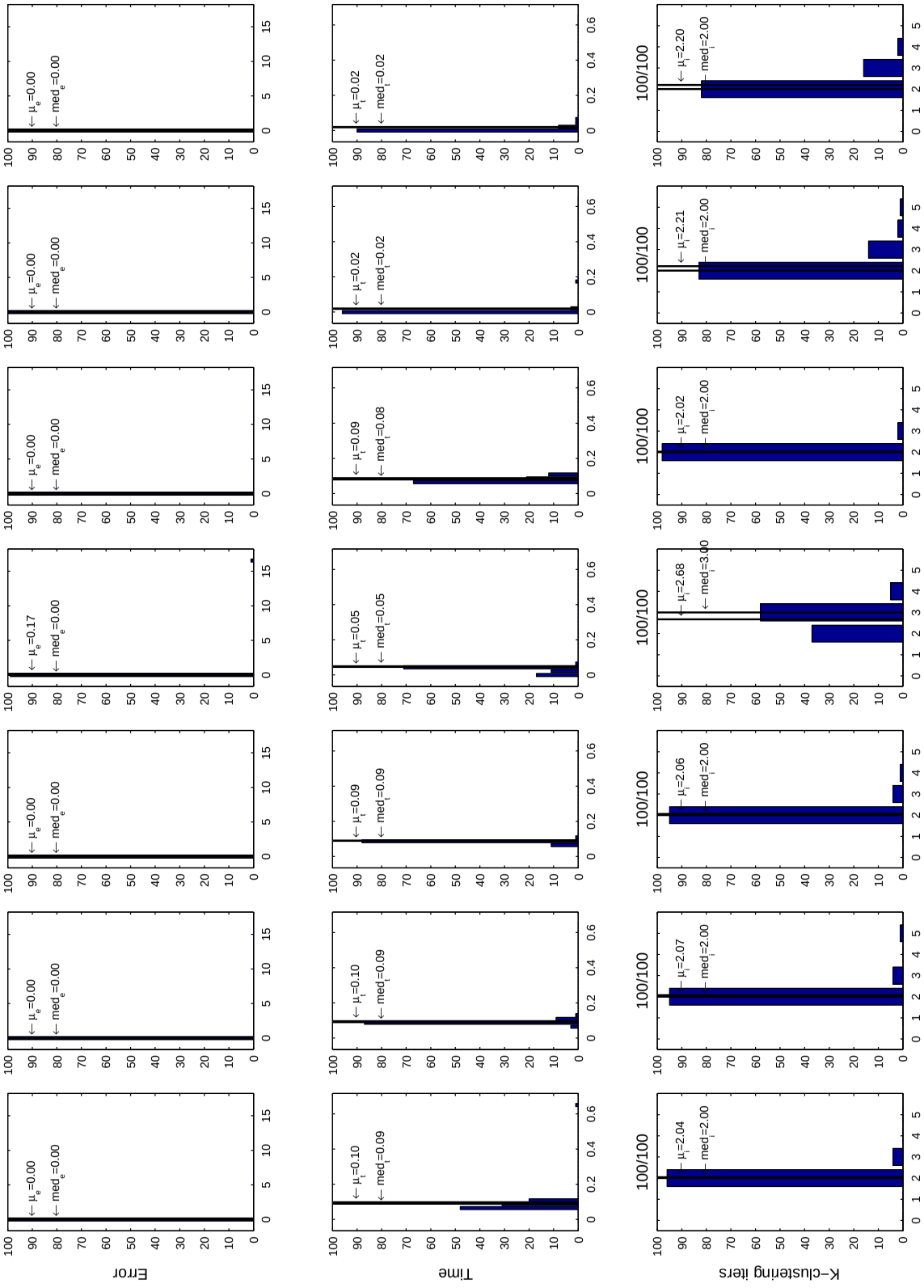


FIGURE 45 Test distributions of 100 test runs. The underlying sampling distribution of the data is a two-mode Gaussian mixture with 15% of missing data. $N_k = 50$ for $k = \{1, 2\}$. The results for the methods are presented column-wise from left to right: BF, TrobBF, robBF, random, BF+K-means, ModKKZ, and TModKKZ.

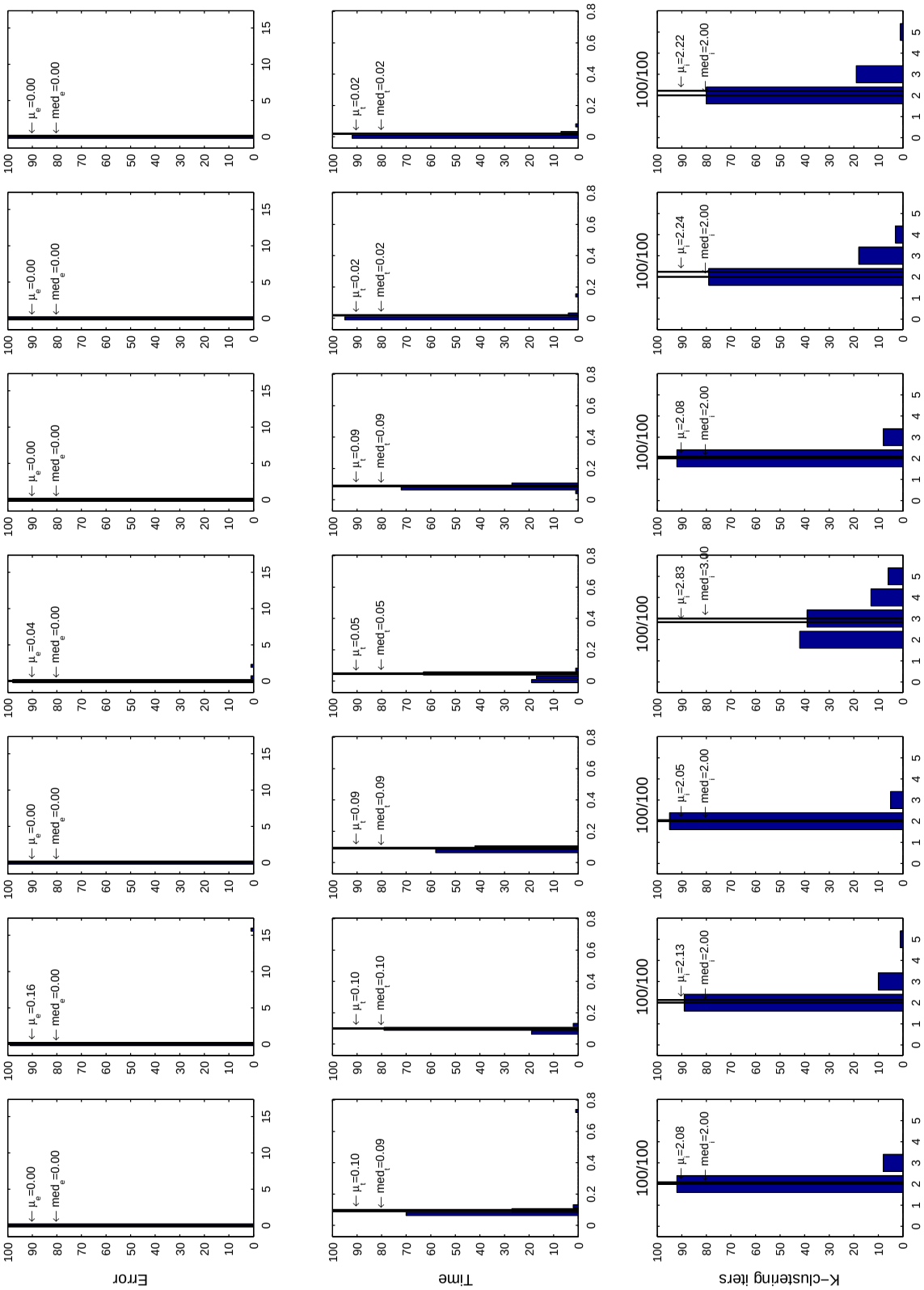


FIGURE 46 Test distributions of 100 test runs. The underlying sampling distribution of the data is a two-mode Gaussian mixture with 45% of missing data. $N_k = 50$ for $k = \{1, 2\}$. The results for the methods are presented column-wise from left to right: BF, TrobBF, robBF, random, BF+K-means, ModKKZ, and TModKKZ.

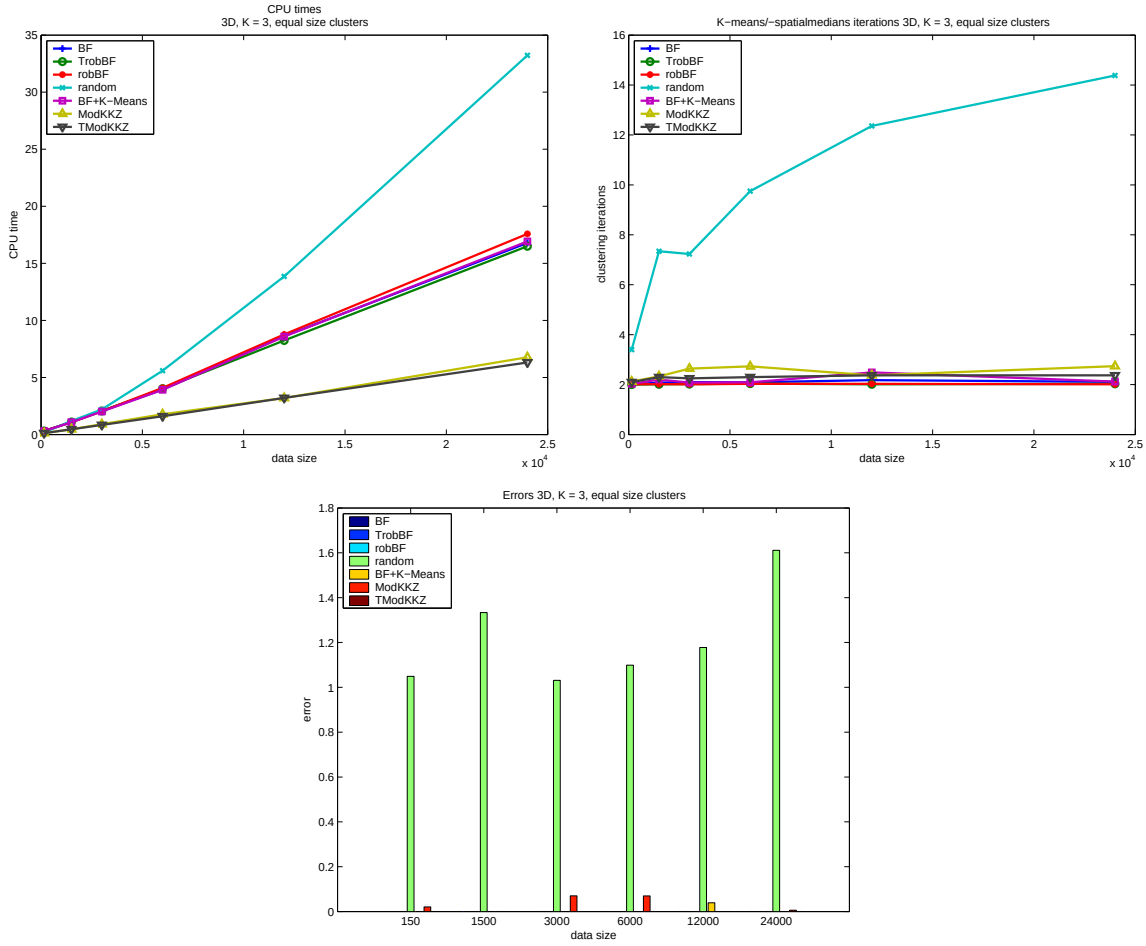


FIGURE 47 CPU time, number of K-means/-spatialmedians clustering iterations, and error divided by the number of clusters and dimensions. In all cases, the clusters are of equal size. Note that the scale of the horizontal coordinates is not linear.

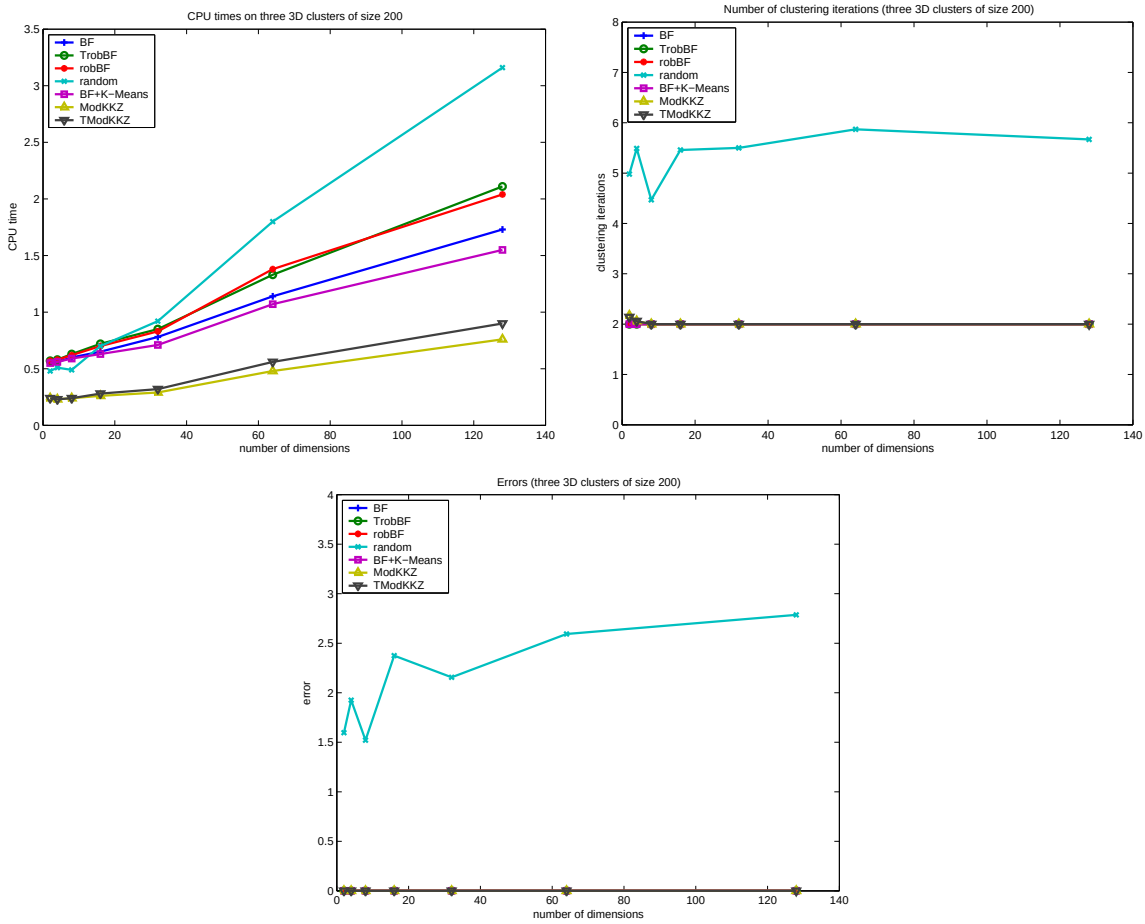


FIGURE 48 CPU time, number of K-means/-spatialmedians clustering iterations needed after the initialization, and error, which is divided by the number of clusters and dimensions. In all cases, the clusters are of equal size.

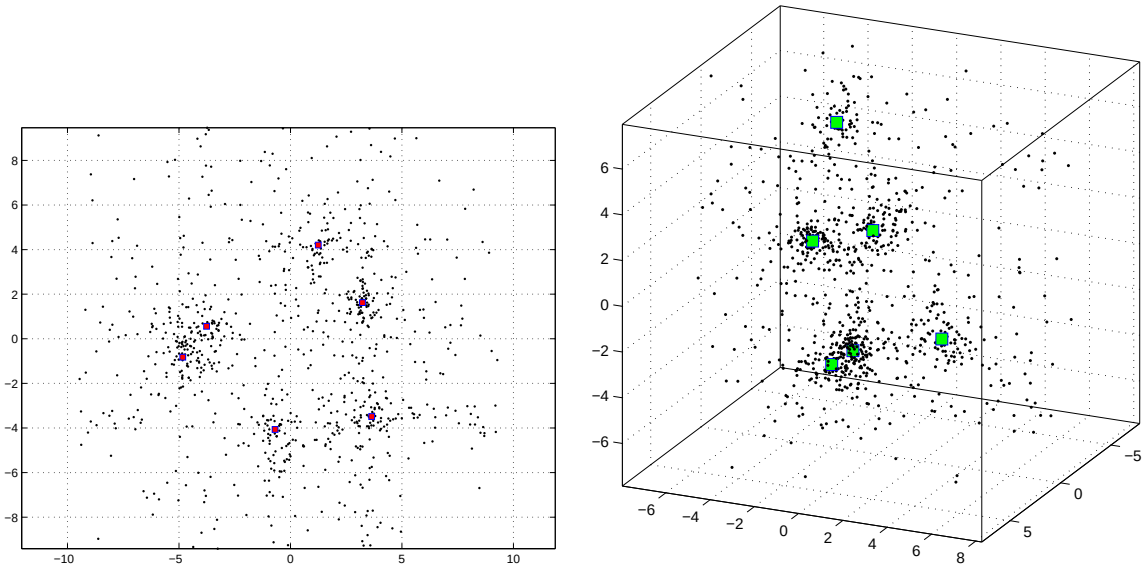


FIGURE 49 2- and 3-dimensional clustered data sets from the Gaussian and Laplace distributions. The data sets contain 30% and 15% of noise, respectively.

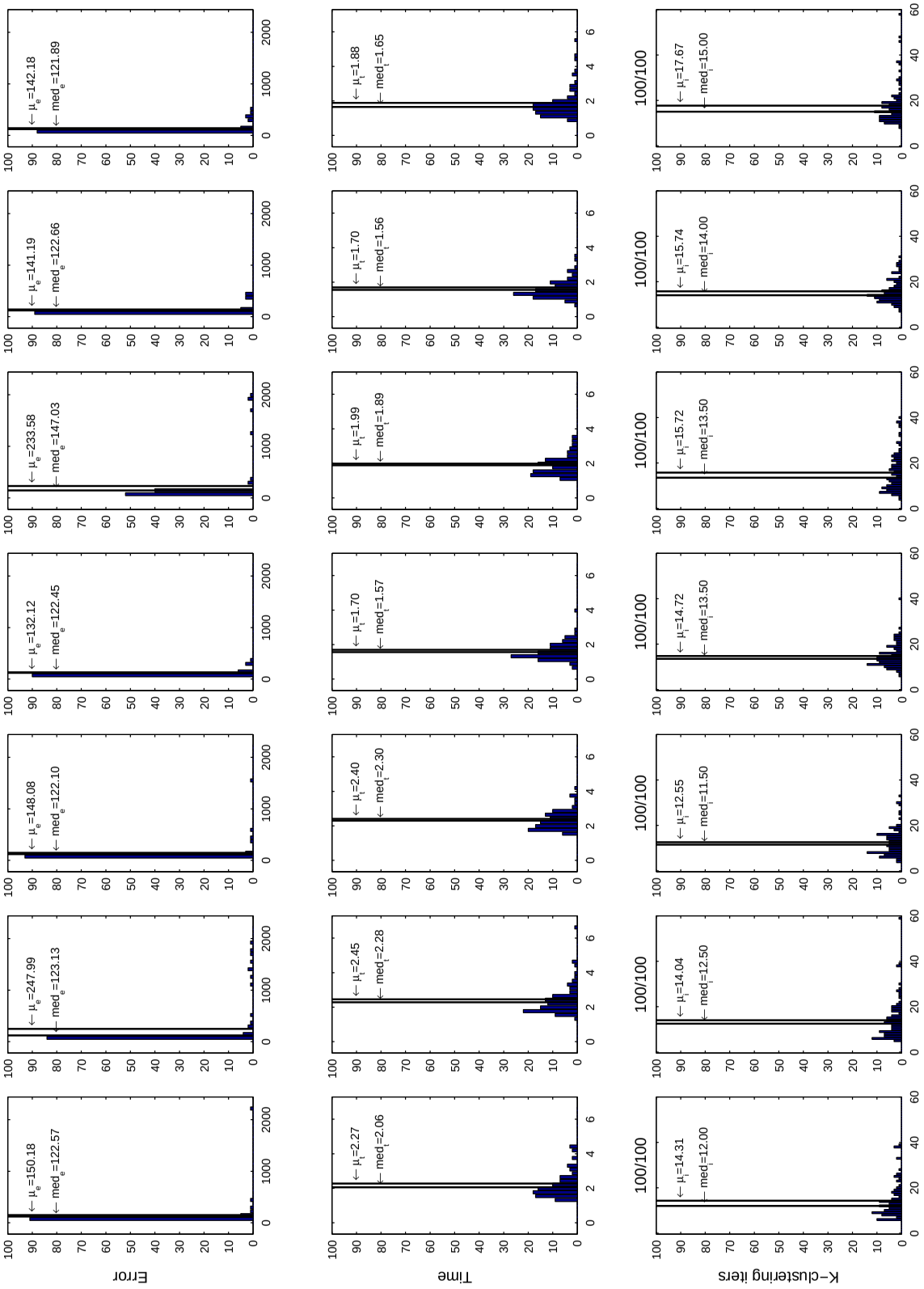


FIGURE 50 3 spherical clusters from a 30D Gaussian and Laplace distribution. The number of data points is 720. Min/max size of clusters is 220/260, noise(%) 30 and missing values (%) 33. The number of the generated data sets is 100. The results for the methods are presented column-wise from left to right: BF, TrobBF, robBF, random, BF+K-means, ModKKZ, and TModKKZ

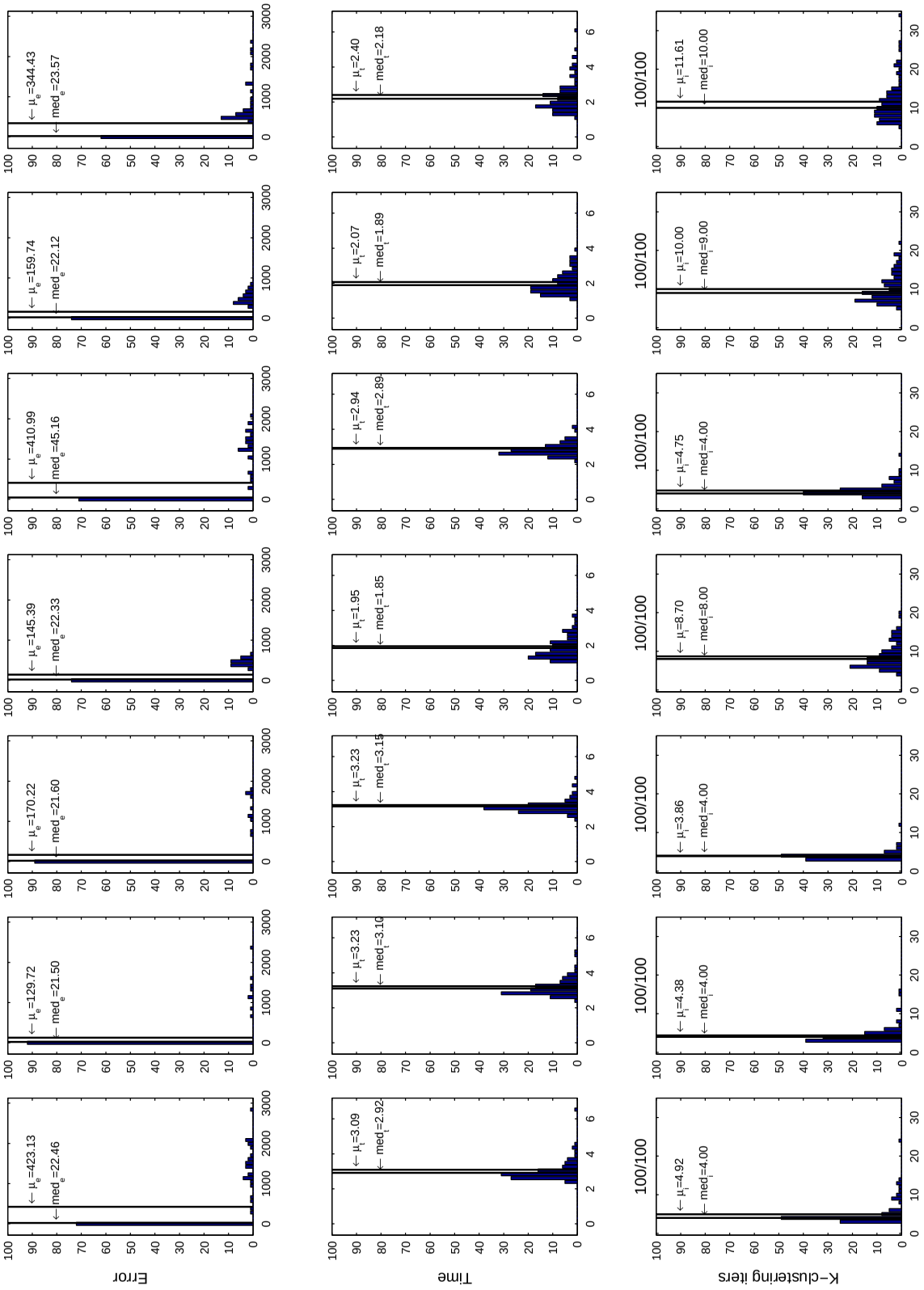


FIGURE 51 6 spherical clusters from a 30D Gaussian and Laplace distribution. The number of data points is 900. Min/max size of clusters is 120/180, noise(%) 10 and missing values (%) 33. The number of the generated data sets is 100. The results for the methods are presented column-wise from left to right: BF, TrobBF, robBF, random, BF+K-means, ModKKZ, and TModKKZ

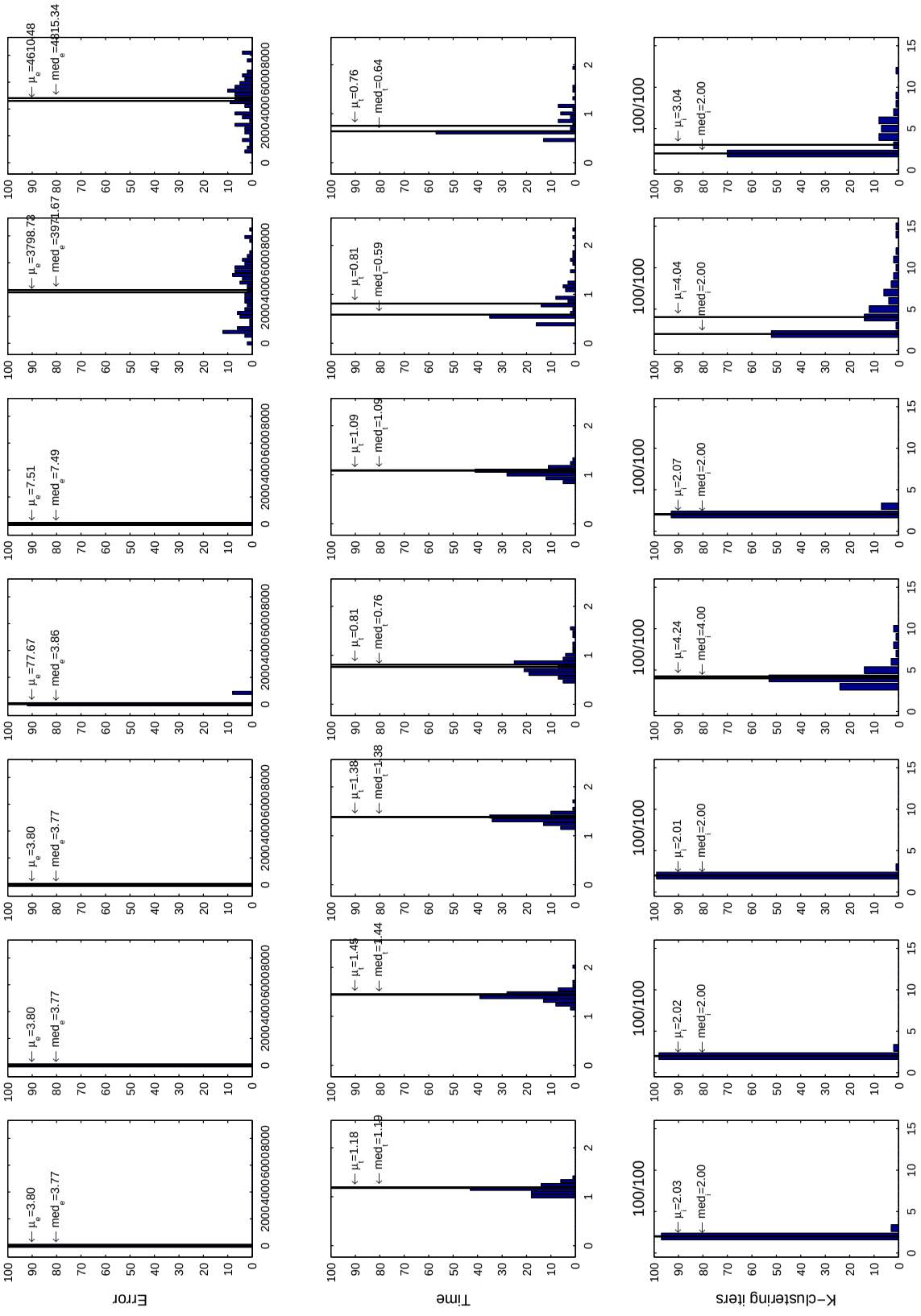


FIGURE 52 3 spherical clusters from a 50D Gaussian and Laplace distribution. The number of data points is 720. Min/max size of clusters is 220/260, noise(%) 5 and missing values (%) 0, 20, and 40. The number of the generated data sets is 100 clusters. The results for the methods are presented column-wise from left to right: BF, TrobBF, robBF, random, BF+K-means, ModKKZ, and TModKKZ

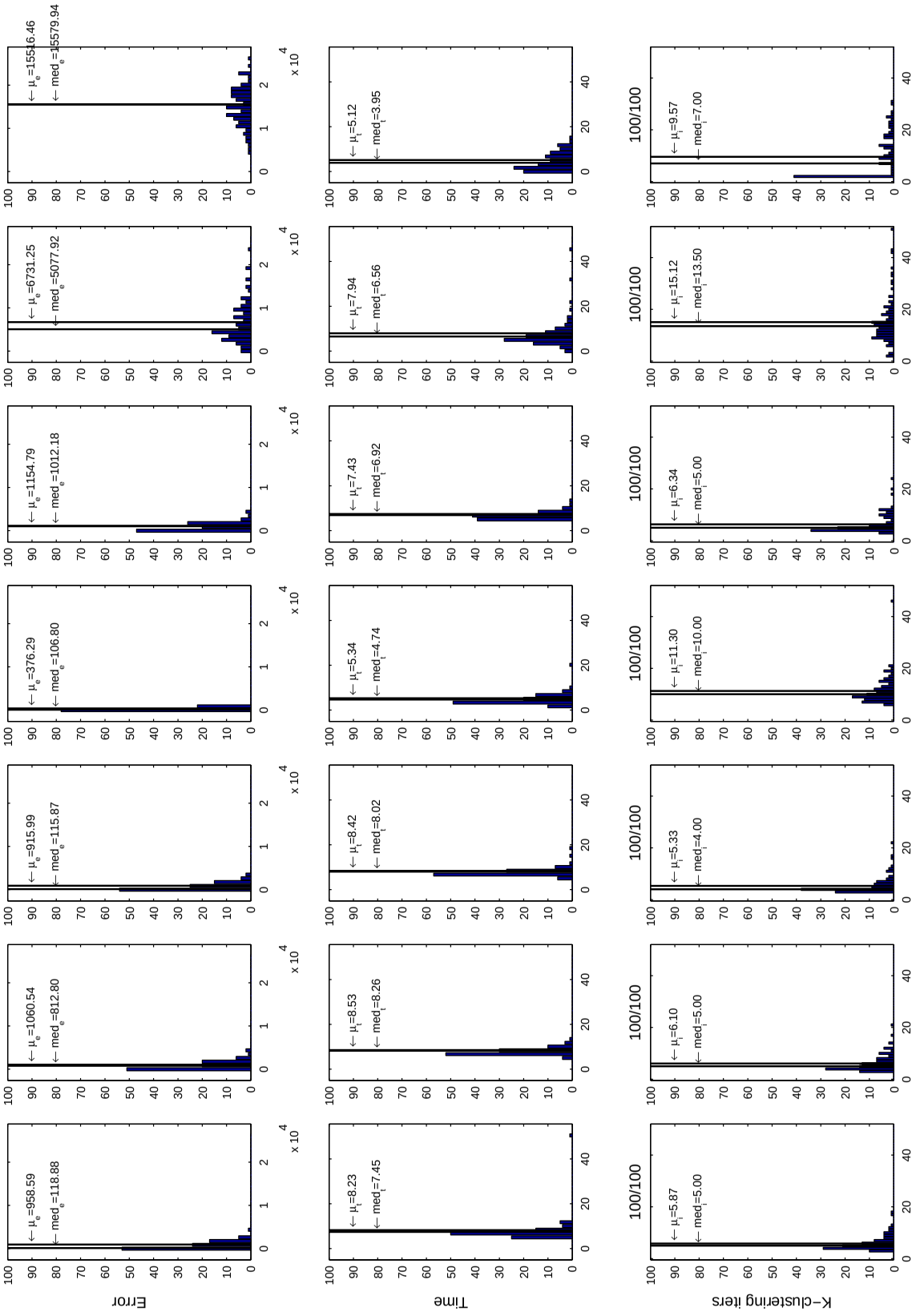


FIGURE 53 7 spherical clusters from a 50D Gaussian and Laplace distribution. The number of data points is 1455. Min/max size of clusters is 100/300, noise(%) 15 and missing values (%) 0, 20, 40, 15, 35, 50, and 5. The number of the generated data sets is 100. The results for the methods are presented column-wise from left to right: BF, TrobBF, robBF, random, BF+K-means, ModKKZ, and TModKKZ

8 MINING REAL APPLICATIONS

In this chapter, the practical utility of the developed clustering methods is demonstrated on a few real world data sets. The obtained clustering solutions are visualized using classical and robust projection techniques. Therefore, a couple of variants for cluster visualization are introduced before their usage in real applications. Furthermore, silhouettes and L1-data depth based indices are presented as promising cluster validity indices. The original silhouettes are replaced with a more robust modification, whose behavior is then compared to the ReD index on real-world data. The chapter does not include thorough tests for any of the proposed methods. The data projection techniques are based on existing robust covariance estimates and they are only applied to data sets and solutions. The cluster validity indices are also based on given suggestions from literature and the robust variant is presented and applied in a tentative manner. As the goal of this chapter is to provide examples of the usability and utility of the developed clustering techniques, thorough testing and analysis (especially knowledge discovery) remains a future challenge. The sample applications involve image compression tasks (vector quantization), analysis of industrial process measurements, and an analysis of data representing information on more than three thousand software projects.

8.1 Dimension reduction and visualization

Dimension reduction and feature selection are closely related to data clustering. The basic principles are introduced in Chapter 3. From the perspective of the DM process, these methods can be used as preparatory tools before the DM step. The dimension reduction may lead, for example, to a faster clustering process. On the other hand, they can be used as explorative methods during the visualization step. When applied to data visualization, dimensionality of the processed data is lowered to enable visual data exploration. The simplest tools for data visualization are, e.g., parallel coordinates and numerous plotting techniques (scatter, trellis, star, box plots, etc.) [170]. The defect of these methods from the DM point of

view is that they are not necessarily able to reveal dependencies and relationships in high dimensional spaces, since the data is not always distributed or discriminated in the direction of individual or pairwise coordinates. Therefore, methods that find some form of principal directions from data are needed. The principal directions can be represented, e.g., by principal components of a covariance matrix, so that data variation is maximized. As the most informative directions are determined for the data, the data points and/or cluster centers can be projected onto these new coordinate axes. In graphical presentations, the first two or three of the most informative axes are chosen as the representative directions of the data.

8.1.1 Robust covariance estimates

Before a more detailed treatment of data projection techniques, some covariance estimates are introduced. These will be applied in the chosen data projection techniques. When the underlying distribution behind the target data is normal, the maximum likelihood estimate of the data variability is defined by the sample covariance matrix (see, Section 4.3.1). Because of the underlying assumptions, the sample covariance is highly sensitive to contaminated data, which means that covariance estimates may break down. This often leads to inflated variances, distorted correlations, and weighting of unnecessary dimensions. By using nonparametric multivariate statistics, robust estimators for covariance matrix have been developed by many statisticians [272, 389, 390]. When compared to the sample covariance estimator, these estimates are inherently more robust against contaminated data (cf. the sample mean versus spatial median in Chapter 4).

Since the sample covariance matrix estimator is already presented in the previous sections, only the robust variants are introduced here. In addition to the classical principal components, the subsequent real-world data sets are visualized by using principal components that are based on nonparametric robust estimation techniques. The first one is the sample sign covariance matrix (SCM) estimate, which is defined as [272, 389]

$$\mathbf{\Sigma}_{SCM} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{S}(\mathbf{x}_i - \mathbf{m})\mathbf{S}^T(\mathbf{x}_i - \mathbf{m}),$$

where $\mathbf{S}(\mathbf{x})$ is *the spatial sign function* that is multivariate generalization of the univariate sign function ($sign(x) \in \{-1, 0, 1\}$, $x \in \mathbb{R}$) defined as [290]

$$\mathbf{S}(\mathbf{x}) = \begin{cases} \frac{\mathbf{x}}{\|\mathbf{x}\|_2}, & \mathbf{x} \neq \mathbf{0}, \\ \mathbf{0}, & \mathbf{x} = \mathbf{0}. \end{cases} \tag{61}$$

This is the spatial sign function that gives the direction of vector $\mathbf{x}$ from the origin. One can see that (61) is a special case of the general non-smooth optimality condition (39) for the problem of the spatial median. The above formulation shows that the SCM based estimators assume that the data is first centered with respect to the spatial median $\mathbf{m}$ of the data. The relative efficiency and robustness of SCM

based estimators is shown to be very comparable to several other covariance estimators by simulation experiments in [272].

The sample Tau covariance matrix (TCM) estimate is defined as [389]

$$\mathbf{\Sigma}_{TCM} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{S}(\mathbf{x}_i - \mathbf{x}_j)\mathbf{S}^T(\mathbf{x}_i - \mathbf{x}_j).$$

The benefit of this estimate is that the spatial median is not needed because it is based on pair-wise directions between the data points. This is, however, a computationally expensive estimator. The estimators based on SCM and TCM are rotation equivariant, but not affine equivariant. Hence, they are sensitive to change of scales.

In order to take into account the robust variability of the data in the direction of the principal component, Visuri et al. [389] propose a special strategy for constructing a robust estimate for the covariance matrix. The eigenvector estimates (denoted by matrix $\mathbf{U}$) are first constructed using a robust procedure, for example, the SCM or TCM estimator. The marginal variances, that are called eigenvalues or principal values and denoted by $\Lambda = diag(\lambda_1, \ldots, \lambda_p)$, of $\mathbf{U}^T\mathbf{x}_1, \ldots, \mathbf{U}^T\mathbf{x}_n$ are then estimated using any univariate robust scale estimate (e.g., median absolute deviation (MAD)). Finally, the covariance matrix estimate is $\mathbf{\Sigma} = \mathbf{U}\Lambda\mathbf{U}^T$.

Principal components

Principal component analysis aims at finding such linear combinations of a data set that preserve the maximum amount of information assuming that the information is measured by variance [272]. Hence, it is natural to use it for explorative data mining. A reduced dimension is obtained when the original high dimensional data is projected from the original $\mathbb{R}^p$ space into the lower dimensional $\mathbb{R}^q$ space ($p \gg q$) that is determined by the principal components.

Let us suppose that the mean $\boldsymbol{\mu}$ of a p-dimensional standardized data set $\mathbf{X}$ is at the origin (otherwise data is first centered to the origin). The q-dimensional projection $\mathbf{y}_i$ of any vector $\mathbf{x}_i \in \mathbf{X}$ is obtained by

$$\mathbf{y}_i = \mathbf{A}\mathbf{x}_i,$$

where $\mathbf{A}$ is $q \times p$ orthogonal matrix. In the classical PCA analysis, the transformation matrix A is defined by the eigenvectors of the covariance matrix $\mathbf{\Sigma}$ of data set $\mathbf{X}$. The eigenvectors $\mathbf{e}_i$ and the corresponding eigenvalues λ_i of the covariance matrix $\mathbf{\Sigma}$ are obtained by

$$\mathbf{\Sigma}\mathbf{e}_i = \lambda_i\mathbf{e}_i, \quad \text{for} \quad i = 1, \ldots, p.$$

The eigenvectors corresponding to the largest eigenvalues represent the directions of the largest variance in the data. By taking some of the largest eigenvectors as the row vectors of $\mathbf{A}$, one obtains a transformation matrix that maps the points of the original space $\mathbb{R}^p$ to the low-dimensional orthogonal representation in $\mathbb{R}^q$.

If data set $\mathbf{X}$ is projected onto this new coordinate system, much information is preserved in the form of variability while the number of dimensions is decreased.

The classical principal components are based on the sample mean and sample (co)variances and correlations. Therefore, they are very sensitive to erroneous data. Consequently, robust covariance matrix estimates, such as the aforementioned sign and rank based SCM and TCM, are used in the principal component analysis. The available case data strategy from Section 3.2.8 can easily be extended to all covariance matrix formulae where missing data exist.

In the case of data clustering, one can also determine the principal component directions with respect to cluster prototypes. This gives many advantages when compared to the use of the full data set. The prototypes are usually complete, which means that missing data is not a problem for the computation of the principal projections. Furthermore, the utilization of prototype data to the computation of the principal projections significantly reduces the computational requirements because usually $K \ll n$. Moreover, if the chosen clustering criterion maximizes the between-cluster distances, the prototype data should contain the most discriminative directions of the data. Hence, by using prototypes in the estimation of the principal directions, the projected data should also contain information about the most discriminative directions of the clustered data that are the directions which maximize the between-cluster distances. Directions of the largest variance do not necessarily guarantee the preservation of the discriminative features to the projected data. Finally, if the cluster prototypes are produced by a robust and reliable clustering algorithm, the principal components are probably better saved from outliers. The risk in the use of the prototype data is that the global structure of data may be more emphasized than the local.

While the principal component mapping is based on the linear projection from high to low dimensional space, multidimensional scaling (MDS) refers to a set of methods that applies non-linear transformation to the data [175, 170]. MDS is also widely used in visual and explorative data analysis.

In MDS, computation is based on the pairwise distances between the data points. Let d_{ij} be the distance (e.g., Euclidean distance) between two p-dimensional observations $\mathbf{x}_i$ and $\mathbf{x}_j$. A standard MDS seeks q-dimensional ($q < p$) vectors $\{\mathbf{x}'_1, \ldots, \mathbf{x}'_n\}$ that minimize the cost function, which is given by

$$\mathcal{J}_{MDS}(\{\mathbf{x}'_1, \ldots, \mathbf{x}'_n\}) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} (d_{ij} - d'_{ij})^2,$$

where d'_{ij} is the distance between unknown q-dimensional vectors $\mathbf{x}'_i$ and $\mathbf{x}'_j$. Hence, the goal is to find a configuration for the data points in $\mathbb{R}^q$ that preserve the pairwise distances of the data vectors as well as possible. The initial configuration of points to the q-dimensional space can be created, e.g., using the PCA method. The cost function is minimized by using an appropriate optimization algorithm.

Some variants of the standard formulation of the MDS problem exist. Per-

haps the most popular is *the Sammon's mapping* that is given by

$$\mathcal{J}_{Sammon}(\{\mathbf{x}'_1,\ldots,\mathbf{x}'_n\}) = \sum_{i=1}^{n}\sum_{j=i+1}^{n}\frac{(d_{ij}-d'_{ij})^2}{d_{ij}}.$$

Due to the normalization by the pairwise distances of the original space, smaller distances are better emphasized. Smaller weighting for large distances could also be obtained by omitting the squaring of the subtraction term. A good side of MDS is that only pairwise distances are required. It also generalizes to any dissimilarity measure. The weakness of MDS is the computational load.

The Sammon's mapping was preliminary tested by us for full data sets and cluster prototypes. Because the mapping did not produce remarkable improvements (i.e., changes) to the principal component projections, it is not applied in the subsequent real-world examples. In these tests, also a relatively large amount of computation time was needed.

Another interesting approach to the cluster projection and visualization is the classical linear discriminant analysis [94]. Based on the Fisher's linear discriminant and its generalization, Dhillon et al. [86] propose a formulation of class-preserving projections for cluster and class visualization. The objective is to maximize the following ratio

$$\frac{tr(\mathbf{W}\boldsymbol{\Sigma}_B\mathbf{W})}{tr(\mathbf{W}\boldsymbol{\Sigma}_W\mathbf{W})},$$

where $\mathbf{W}$ represents the most discriminant low-dimensional orthonormal basis and $\boldsymbol{\Sigma}_B$ and $\boldsymbol{\Sigma}_W$ are the between-scatter and within-scatter matrices. Dhillon et al. [86] ignore the within-cluster scatters $\boldsymbol{\Sigma}_W$ in order to reveal the multidimensional class-structure from the data.

8.2 Data-based indices for the correct number of clusters

Cluster validation is an interesting and challenging problem. The traditional iterative relocation methods, such as K-means, do not provide information about the number of clusters K. It is clear that the sum of the squared error criterion monotonically decreases as K increases and reaches its minimum when $K = n$ (e.g., Duda et al. [93, p.241]). Hence, this favors small clusters and is of no use *per se* for cluster validation. Usually validation indices measure compactness and separability of clusters. There are several approaches to measure the inter-cluster distance [156]:

- Single linkage that measures the distance between the closest members of the clusters.
- Complete linkage that measures the distance between the most distant members of the clusters

- The distance between the cluster prototypes.

Moreover, based on experimental results, Bezdek et al. [35] emphasize that inter-cluster separation has a more important role in cluster validation than within-cluster scatter.

Consequently, a number of internal and external indices exist for cluster validation and for the problem of choosing the correct number of clusters (see, e.g., [283, 157]). External indices are based on test data sets on which the obtained clustering solutions are validated. Internal indices are typically based on the between- and within-cluster distances. A third approach is to use a relative criterion that compares the results of different clusterings obtained with the same algorithm, but with different parameter settings [158]. Sometimes the heuristic for the problem of unknown number of clusters is integrated to the clustering algorithm. Furthermore, also visual validation methods have been introduced. An extensive review on the methods and indices can be found in [156].

Estimation of the correct number of clusters is a difficult task. For a thorough examination of different indices, several different clustered data sets, experimental settings, and clustering algorithms must be considered (see, e.g., [283, 91, 173]). Moreover, the results of data clustering are always somewhat data dependent. Here, a new variant for an existing index is proposed. The new index is applied to the real-world data sets in this chapter, but not thoroughly tested yet. The results of the new index are shown by graphical curves that are expected to give a clue for the analysts about the most appropriate and "natural" cluster models. These can (and preferably should) then be compared to the models considered best by the visual "human validation". Hence, when applying these computational indices one should recall the general nature of the cluster analysis process explained in Chapter 3, which should not be structure imposing, but rather let the data tell about itself. Before introducing the methods used, a brief review of the existing methods is given.

Milligan et al. [283] investigate the performance of 30 procedures for estimating the number of clusters. The best method found in their study is the *Calinski and Harabasz index* (CH) defined for n data points and K clusters as

$$\frac{\mathrm{tr}(\mathbf{\Sigma}_B)/(K-1)}{\mathrm{tr}(\mathbf{\Sigma}_W)/(n-K)},$$

where $\mathbf{\Sigma}_B$ and $\mathbf{\Sigma}_W$ are the between cluster scatter matrix and the sum of the within cluster scatter matrices, respectively. CH performs consistently across data sets with varying numbers of clusters. The second best index is the ratio criterion $\mathcal{J}_e(2)/\mathcal{J}_e(1)$ introduced by Duda and Hart [93]. $\mathcal{J}_e(1)$ and $\mathcal{J}_e(2)$ are the sum of squared error criterions for one and two clusters, respectively. A predetermined critical value for the index is used for deciding whether or not the splitting of a cluster is justified. Although this index has some difficulties when the true number of clusters is small ($K = 2$), its performance is otherwise comparable to CH. The examination showed poor performance of the traditional indices, $\mathrm{tr}(\mathbf{\Sigma}_W)$,$\mathrm{tr}(\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B)$, and $|\mathbf{T}|/\mathbf{\Sigma}_W$ ($\mathbf{T} = \mathbf{\Sigma}_W + \mathbf{\Sigma}_B$) that assume multivariate normality (see, [130]).

Dubes [91] presents the results of thorough experiments for two internal validation indices called the *Davies and Bouldin index* (DB) and *modified Hubert* Γ *statistics* (MH). As a result, the latter is shown to perform better under all experimental conditions.

Hardy [173] presents good results in estimating the true number of clusters for three methods based on hyper-volume criterion. He compares these methods against four other indices, such as the well-known Marriot's $k^2|\mathbf{\Sigma}_W|$ index [275], that has been available in Clustan[1] cluster analysis software. As a result, Hardy recommends the application of several clustering strategies and validation indices to learn more about the clusters and use all this information.

Bezdek et al. [35] show that the so-called *Dunn's index* is very sensitive to noisy data points. They introduce 17 generalized variants for the Dunn's index and compare those against DB, MH, and the original Dunn's index. They find that the Dunn's index in its original form is not a suitable for cluster validation, but it can be enhanced with a proper modifications. As a conclusion, they suggest, similarly to Hardy [173], to use several clustering methods and strategies, vary the parameter settings and collect many "votes" using various indices. Consistent results across various trials indicate that the true number of clusters is found.

Halkidi et al.[160] propose the SD index that measures the weighted sum of the average within-cluster scattering and the total inter-cluster separation. Another validity index called S_Dbw by Halkidi et al. [159] is based on the principle that the density of points between clusters should be low in comparison with the intra-cluster density. Low inter-cluster density indicates well-separated clusters. Hence, the S_Dbw index measures the sum of the inter-cluster density and within cluster scattering.

Maulik et al. [277] introduce a new validity index $\mathcal{I}$ and compare it against the DB, CH, and Dunn's indices. The $\mathcal{I}$ index performs best on each of the test cases. The CH index is the second best in this test.

Kim et al. [225] proposed recently a subdivision of cluster validation indices into *summation type* and *ratio type indices*. The idea is to separate the coupling of inter- and intra-cluster distances to each other. They introduce six new alternatives for DB, SD, S_DBw, and two fuzzy indices, and compare the new indices against the old ones. The results show that the modifications lead to improvements and generally ratio-based indices show better performance than summation based indices.

Nakamura et al. [293] introduce an algorithm that simultaneously computes the number and locations of cluster prototypes. The method is called Multi-scale clustering (MSC). The idea is to vary the problem resolution by changing a special scale parameter. The correct number of clusters is defined to be the one that tolerates the largest changes of the scale parameter, in other words, has the longest lifetime.

Visual validation techniques for the cluster analysis are proposed by Ling [256], Bezdek et al. [34], Hathaway et al. [177], and Huband et al. [194]. Instead of

[1] http://www.clustan.com/

using a single number as a measure of cluster validity, they apply dissimilarities, grids, and shades of gray to the validation.

Hamerly et al. [161] propose the Gaussian-means algorithm that starts with a small number of clusters. The number of clusters is increased after each iteration by splitting the clusters which do not satisfy the Gaussian assumption according to a statistical test. The algorithm stops after the hypothesis that the data assigned to the cluster is Gaussian is accepted. The method outperforms the X-means algorithm [314] based on the *Bayesian information criterion* (BIC) on data with non-spherical clusters. X-means tends to overestimate the number of clusters on such data. Another example of the BIC criterion based methods with multivariate normal mixture model clustering is introduced by Fraley et al. [126].

Bischof et al. [37] introduce a robust clustering algorithm whose validation is based on the *Minimum Description Length* (MDL) criterion.

Smyth [353] introduces a clustering algorithm that is based on the *Monte Carlo Cross-validation*. He applies two approaches known as the MCCV and v-fold cross-validation. The performance of the MCCV method is roughly comparable with the AutoClass method [67] and slightly better than the BIC criterion based clustering. The v-fold cross validation method shows unreliability.

Tibshirani et al. [371] introduce the *Gap-statistic* method, which compares the curve of $\log W_k$ (W_k is the total sum of within-cluster distances) to the curve obtained from data uniformly distributed over a smallest hyper-rectangle containing the data. The optimal number of clusters is obtained when the gap between the curves is largest.

Prediction strength method is proposed by Tibshirani et al. [370]. This method splits data in two halves and runs the clustering algorithm for both. Then, applying the nearest-centroid principle, the other half, test data, is classified using the prototypes obtained with training data. The similarity of the clusterings is measured as the average of co-memberships in the worst matching clusters. The clustering that best predicts the classification is selected.

Levine et al. [250] introduce a resampling-based method for estimating the correct number of clusters for data. The idea is to assess the average similarity between solutions that are computed for the full data and for a given number of subsamples of size fn, where $f \in [0,1]$. The merit $\mathcal{M}(\mathcal{K})$ for each clustering is calculated. It measures the agreement between the clusterings obtained on the sub-sample and full data. $\mathcal{M}(\mathcal{K}) = 1$ means perfect agreement. The clustering with the highest value of the merit $\mathcal{M}(\mathcal{K})$ is chosen to be the best solution.

Roth et al. [336] and Lange et al. [243] present a resampling approach to cluster validation that is based on the concept of cluster stability. The basic idea behind the method is again to split data into two halves and apply the clustering algorithm to both. Then the other half is used to train a classifier and predict the classes of the remaining half of the data. The distances of the solutions obtained for the latter half of data are computed as misclassification rate. K that yields the smallest average misclassification rate is chosen as the number of clusters.

Dudoit et al. [95] introduce a prediction-based sampling method, CLEST, for estimating the number of clusters. In the CLEST procedure, the data is first

split into two non-overlapping sets. The learning set is clustered and a classifier is built using the obtained labels and applied to the test set. The test set is then also clustered and the obtained labels are then compared using an external index (the authors suggest the FM index [122]). This is repeated perhaps 20 times and the median estimate for the similarity statistics is computed. The same procedure is repeated for the data sets generated under null hypothesis. The statistics are then compared in order to decide the number of clusters.

In summary, many attempts to invent a reliable validity index for clustering problems have been proposed. The earliest methods have mainly been internal indices. Later, the collection of validity indices have diversified as computationally intensive variants, such as cross-validation, BIC, MDL, resampling, stability, and even visual methods have become more common. Although it is difficult to find the best index, one may recognize some weak indices, such as the Dunn's index. Hence, a good index should perform well at least under well-defined cluster structures [106, p.103]. To obtain the best result, one should perhaps take several indices, parameter settings, and clustering algorithms, and then perform "voting" to find the largest agreement between the methods [173, 35, 143, 106]. On the other hand, one may also get bad results if the voting is based on wrong assumptions (concerning normality, missing data, outliers, etc.)

8.2.1 Silhouettes

Silhouettes [220] and ReD [207] are the chosen validity indices in this chapter. Actually, the first one is made robust by a trimming technique, while the second one is used in its original form. These are simple data-based internal indices. ReD is initially based on robust estimates and Silhouettes can be easily made more robust by trimming.

Silhouette width is an internal cluster validation approach that measures the cluster tightness and separation [337, 220]. The silhouettes can be used for graphical illustration of the clustering solution. Here the principle is used as an internal validation index.

Let us next consider how the silhouette values are computed for a partitioned data set $\mathbf{X}$. Let $a(\mathbf{x}_i)$ be the average dissimilarity (usually with respect to the Euclidean distance) of $\mathbf{x}_i \in X$ to all other objects in the same cluster. Furthermore, let $\mathcal{C}$ be the second closest cluster to $\mathbf{x}_i$. Then $b(\mathbf{x}_i)$ denotes the dissimilarity from $\mathbf{x}_i$ to $\mathcal{C}$, which is defined as the average dissimilarity from $\mathbf{x}_i$ to the objects in cluster $\mathcal{C}$. The silhouette width $s(\mathbf{x}_i)$ is then calculated by

$$s(\mathbf{x}_i) = \frac{b(\mathbf{x}_i) - a(\mathbf{x}_i)}{\max\{a(\mathbf{x}_i), b(\mathbf{x}_i)\}}. \tag{62}$$

If the cluster consists of only a single object, the definition of $a(\mathbf{x}_i)$ is unclear. Kaufman et al. [220, p.85] suggest to set $s(\mathbf{x}_i) = 0$. The average silhouette width for k cluster solution is known as the *silhouette coefficient* [220]

$$\bar{s}(k) = \frac{1}{n} \sum_{i=1}^{n} s(\mathbf{x}_i).$$

Hence, the "best" number of clusters is obtained by

$$\arg\max_{k=1,\ldots,n-1} \bar{s}(k).$$

Practically, the maximum value of k is much less than $n-1$ in real-world applications.

8.2.2 ReD

A related index to silhouettes is the ReD selection index [207]. It is an internal cluster validation index, which measures the difference between the within- and between-group data depths. As a rank-based method, ReD is independent on the scales of the clusters and is not dominated by high variance clusters.

The ReD index is based on the L_1-*data depth* function, which is introduced using the spatial median [382]. Let us next present the ReD validity index for a partitioned data set X according to [207].

Based on (61), the spatial rank of point $\mathbf{z} \in \mathbb{R}^p$ is defined as [290]

$$R(\mathbf{x}_i - \mathbf{z}) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{S}(\mathbf{x}_i - \mathbf{z}). \tag{63}$$

The data depth of $\mathbf{z}$ with respect to data $\mathbf{X}$ is then defined by [382, 207]

$$D(\mathbf{z}) = 1 - \max(0, \|R(\mathbf{x}_i - \mathbf{z})\| - f(\mathbf{z})), \tag{64}$$

where

$$f(\mathbf{z}) = \frac{\sum_{i=1}^{n} I(\mathbf{z} = \mathbf{x}_i)}{n}.$$

$I(\mathbf{z} = \mathbf{x}_i)$ gives the number of points $\mathbf{x}_i \in \mathbf{X}$ that overlap with $\mathbf{z}$. The statistical interpretation of the data depth relies on the observation that $1 - D(\mathbf{z})$ is the minimum additional weight needed at point $\mathbf{z}$ to make it the spatial median of $\{\mathbf{z}\} \cup \{\mathbf{x}_i\}_{i=1}^{n}$ [207].

Let us denote the data depth of point $\mathbf{x}_i$ with respect to cluster $\mathcal{C}_k$ by $D(\mathbf{x}_i|k)$. The size of cluster $\mathcal{C}_k$ is denoted by n_k. The cluster-wise data depths are made comparable by normalizing them as

$$D(\mathbf{z}|k) \leftarrow \frac{n_k D(\mathbf{z}|k)}{\sum_{i=1}^{n_k} D(\mathbf{x}_i|k)}, \quad \text{for all} \quad \mathbf{z} \in \mathbb{R}^p. \tag{65}$$

From this it follows that the average data depth is 1 for each cluster.

Let us denote the within-cluster data-depth of data points $\mathbf{x}_i \in \mathcal{C}_k$ as $D_i^w = D(\mathbf{x}_i|k)$. The between-cluster data-depth of point $\mathbf{x}_i \in \mathcal{C}_k$ is defined as $D_i^b = D(\mathbf{x}_i|k')$ where k' is the second closest cluster to $\mathbf{x}_i$. An observation $\mathbf{x}_i$ is well-clustered if $D_i^w \gg D_i^b$. Hence, the ReD validation index is defined by

$$ReD(K) = \frac{\sum_{i=1}^{n} ReD_i}{n}, \tag{66}$$

where

$$ReD_i = D_i^w - D_i^b. \tag{67}$$

The number of clusters K is the one that maximizes $ReD(K)$.

8.2.3 Trimmed Silhouettes

Instead of using the original silhouette index, a trimmed version is proposed and later applied in Section 8.3. This is a new approach to compute the index for cluster validation, as far as the author is aware. The idea is based on the assumption that up to a half of the real-world data sets may consists of noise and outliers. Even if the clusters were relatively distant to each other, there may exist some noise and outliers far from the clusters. These might influence significantly the value of the index, because even a small bunch of outliers far from the cluster center increases the average of the within cluster distances considerably. However, it may not be necessary to create a new cluster each time such an outlying set of points occur.

The original silhouettes are based on the average of the Euclidean distances and, therefore, they are very sensitive to outliers. Actually, one outlying data point can break down the average silhouette value of otherwise compact cluster structures. Hence, it might be enough to concentrate the validation on the core of the data clusters. This follows the principles of trimming of data clusters [11].

The computation of a trimmed silhouette is started by finding the cluster-wise closest 50% of the data with respect to the prototype (the spatial median of the cluster data). If missing data exist, general distance measure (6) is applied to the distance computations in order to make the distances from the points to the cluster centers mutually comparable. Thereafter, a pre-determined fraction (at maximum 50%) of the data is removed cluster-wise so that only the most central points are left for the index computation. The average distance $a(\mathcal{C}_k)$ from the most central points to the closest cluster center $\mathbf{m}_k$ is then computed. Hence, $a(\mathcal{C}_k)$ gives information about the tightness of the cluster k. Similarly, the average of the distances from each point of the most central ones is computed to the second closest (neighbor) cluster centers $(\min_{k \neq k'}(d(\mathbf{x}_i, \mathcal{C}'_k)))$ and this is denoted by $b(\mathcal{C}_k)$. The *robust silhouette width* for cluster k is then defined as

$$s_r(\mathcal{C}_k) = \frac{b(\mathcal{C}_k) - a(\mathcal{C}_k)}{\max\{a(\mathcal{C}_k), b(\mathcal{C}_k)\}}.$$

The "best guess" of K is the one that yields the largest value for the *robust silhouette coefficient*, which is defined by

$$\bar{s}_r(K) = \frac{1}{K} \sum_{i=1}^{K} s(\mathcal{C}_i).$$

This approach is utilized in the real-world examples and compared to the robust ReD index. More thorough statistical tests are left for future efforts.

8.3 Real-world applications

Three quite different real-world sample applications are presented to illustrate the usability of the introduced methods.

In the first example, a sample data set from paper industry is clustered and analyzed. Large industrial processes are a good example for the use of DM methods, because huge amounts of data are collected in many different formats. This data contains a lot of useful and valuable information for process control and production management. In the second example, a couple of standard test images are quantized by clustering. Color images are inherently of large size and, thereby, serve as an excellent example for DM methods that are intended to large-scale problems. These types of image clustering tasks are better known as vector quantization and signal compression [140, 175]. The third example is focused on the software project data. The data set used describes various properties from 3024 software projects. The data set is very sparse, which makes it extremely challenging for any data clustering or mining approach.

Based on the experiments and results of this thesis, a new spatial median based clustering method, K-spatialmedians, together with the robust refinement initialization method robBF and the two aforementioned heuristics for estimating the "best number" of clusters, are used for clustering all the aforementioned data sets.

In this chapter, all the examples are realized using the MATLAB 6.1. environment with self-implemented codes. Only the very basic MATLAB library functions were utilized, while the others were implemented using MATLAB macros or C-language and mex-gateway interface. The hardware consisted of a usual PC-computer system with 1.4 GHz AMD Atlon processor, 256 MB of RAM, and the Windows XP professional operating system.

In all cases, the variables and observations that contain only missing values were eliminated during the preprocessing step. In the data transformation step, all variables were shifted and scaled to the closed interval $[0, 1]$. After these operations, the data were clustered by the K-spatialmedians clustering algorithm using several values for K. Each start of the K-spatialmedian algorithm was initialized by the result of the RobBF method. RobBF used 10 sub-datasets for the refinement. Trimming was not used in any of the algorithms. The number of clusters was estimated by using the proposed validation indices.

Three different covariance matrix estimates were used for principal component estimation. Classical sample and sign covariance matrices were computed and two eigenvectors having the largest eigenvalues were used as estimates of the principal directions. Tau covariance matrix, TCM, was combined with the robust estimates of scale as described in Section 8.1.1. Furthermore, some application specific visualization techniques were also used in order to improve the interpretation of the results.

8.3.1 Paper industry

This example illustrates the utility of the proposed methods in analyzing large industrial processes. In reality, capability and possibilities of DM and KM are much broader in the process industry than this example shows. The main intention of the example is to study the functionality and usability of the methods and

techniques in the context.

Paper is an outcome of a large production process, where an enormous amount of mineable data is produced and collected. This data includes several facts, for example, values of physical and chemical measurements that are continuously collected automatically or manually from the pulp and paper making machine. On the other hand, a lot of settings and parameters are configured and adjusted so that the process remains stable. Information about these settings provides valuable information for later analyzes. Furthermore, a number of on-line and off-line laboratory measurements are collected about the quality of the produced paper. This helps the operators to adjust the process. Finally, a lot of human knowledge related to the paper making process is transferred, and some amount of this is, or at least should be, digitized to directly mineable formats. The knowledge about the process exists in many forms, such as reports, e-diaries, e-mails, instructions, html-pages, educational material, electronic customer feedback, etc. In order to enhance and control this kind of industrial processes and, especially, to maximize profit, all related persons should be able to refine, compress, and convert the most valuable and significant information to knowledge from the available data.

In order to keep the focus on technical issues, a minor example by using the proposed clustering and visualization techniques is given next[2]. The KM process was carried out so that the interesting data sets were first selected by domain experts. In this work they were supported by information system specialists who investigated information and communication flows in organization by using the genre-based analysis approach (see Section 2.3.1). The target data is simply called here 'paper data'.

The mining process including the tasks from storage type conversions to visual presentations was then performed by the author. At the same time new algorithms were tested and compared with more traditional methods.

Finally, the graphical interpretation and evaluation of the results was performed together with the domain experts. Due to the confidential nature of the application, technical parts of the proposed KM model are discussed without going into detailed interpretations and analysis of the results and their meanings.

The target data set includes 804 measurement points that represent the state of the process during a continuous period of time. A single measurement point produces information about 53 physical and chemical quantities in this case. These are measured both on-line and off-line from the pulp and completed paper. Hence, the size ($n \times p$) of the processed data matrix is 804×53. The data set provides an excellent example for data mining techniques, because such high dimensional and numerous data matrices are too complicated for explorative analysis without partitioning and refinement operations. On the other hand, more data would not further benefit this example.

---

[2] This example is based on real-world data sets that have been received from Finnish forest and paper industry companies UPM-Kymmene and Metso Paper. The author of this thesis has been a member of a research group that has attended many applied research projects [211, 210, 212] in collaboration with domain experts from the partner companies.

The missing portion of the data is 26,9%, which means that the missing data treatment is necessary. The preprocessing step revealed that four out of the 53 attributes and 20 out of 804 observations were represented by empty values and consequently they were removed before any further processing. In the data transformation all columns of the data matrix were scaled to the range $[0,1]$. Hence, the actual size of the clustered data matrix was 784×49 with approximately 18.9% of empty values.

Results

$K = 7$ was chosen to be the maximum number of clusters. The overall time required by the clustering method for all $K = 2,\ldots,7$ was measured in seconds. The robust silhouette and ReD index for each K was computed. The computation of the robust silhouette index took clearly less time than the computation of ReD.

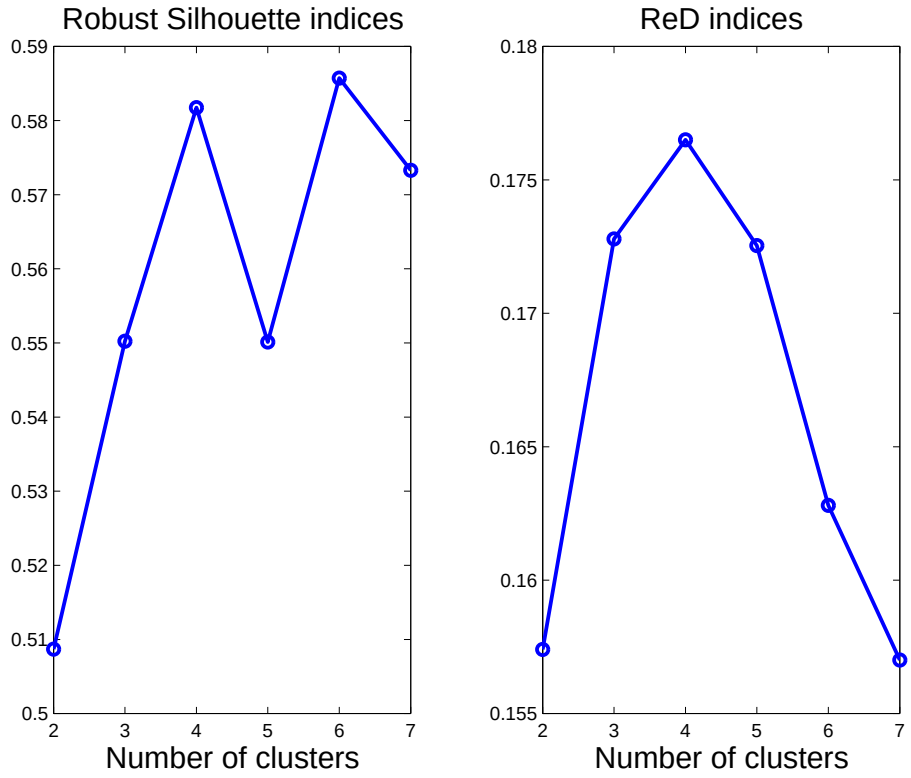


FIGURE 54 Indices for the number of clusters on paper industry example.

Figure 54 shows the values of the two indices for $K = 2,\ldots,7$. The robust silhouette index indicates that data contains four or six clusters while the ReD index shows clearly the four cluster structure. These alternatives will be considered more precisely later, but let us first concentrate on the two cluster case ($K = 2$). Figure 55 presents two-dimensional projections of the paper data with two clusters. All projections suggests that despite the index values in the case $K = 2$, there actually may exist two reasonable groups in the data. The robust projections do not give significant improvements when compared to the classical PCA

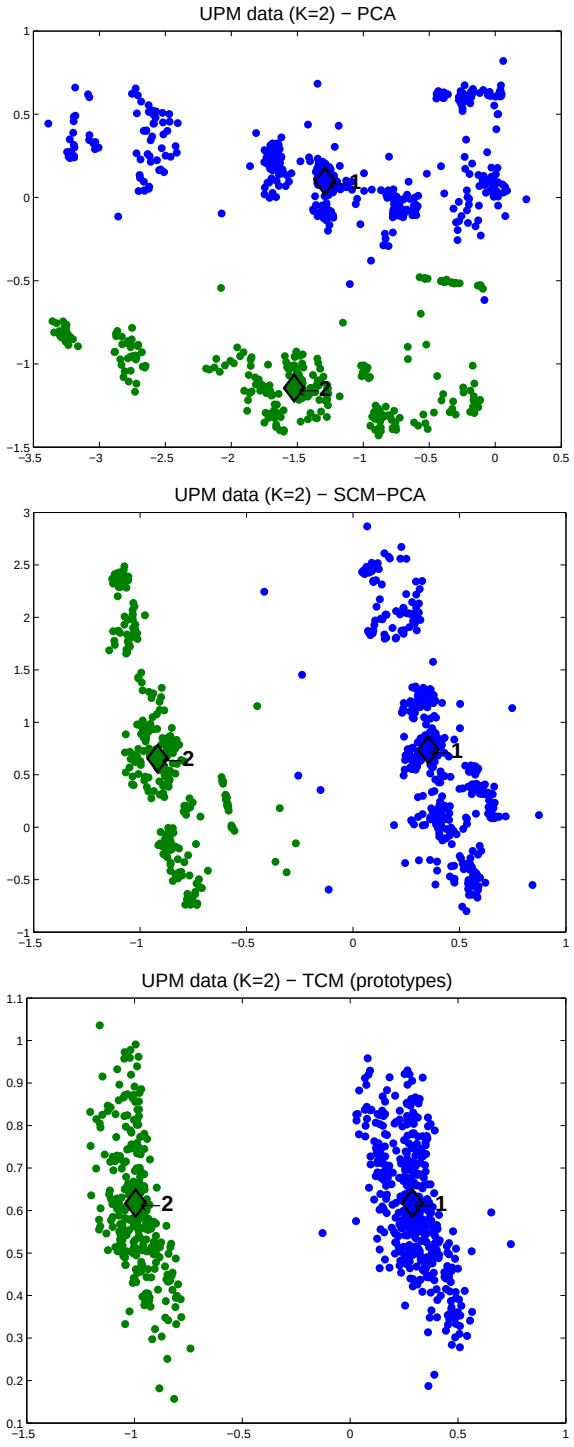


FIGURE 55 The classical, SCM and TCM based principal component projections for 'paper data' in case $K = 2$.

projection. The TCM based projection gives the most compact presentation for the two clusters while the classical PCA projection yields somewhat disordered view. However, it seems that there are no extreme outliers in the data, because the variability of the projections has not inflated in either direction, which may happen for non-robust eigenvalue estimates in the presence of extreme values.

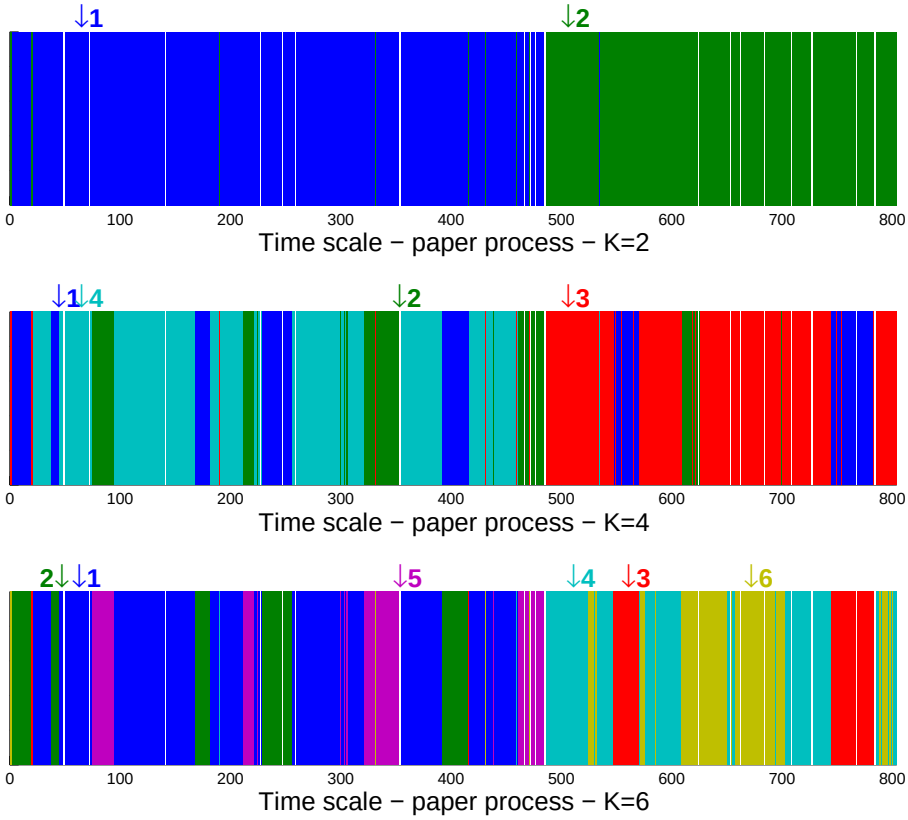


FIGURE 56 Cluster positions in time.

Figure 56 shows the temporal orders of the cluster members on time axis. The absolute times of the measurement points are not presented, but the orders follow the progress of the process. The empty measurement points are represented by white color. Hence, one can easily recognize the points where, for one reason or another, all data are lost. This may be of interest for domain specialists. The graphics clearly shows that two cluster structure splits the process into two almost non-overlapping temporal periods. Points that are the closest in the data space to their cluster centers (the spatial median) are represented by the corresponding numbers and colors. Hence, the clustering indicates that the process period includes two major states, which, moreover, occur in temporally almost continuous periods. Interestingly, the most central points for the process states show up at the early phase of the states. For both states, there are only a few points that clutter up the strict temporal distinction of the states.

The two dimensional projections in Figure 55 reveal well all cluster assignments as well as the approximations for their relative distances. The Euclidean distance of each data point to its closest prototype, measured in the original 53-dimensional data space, is illustrated on the time axis using the scale of gray color in Figure 57. The darker the shade of gray, the closer the point is to its cluster prototype. Previously, for example, Ling [256], Bezdek et al. [34], and Hathaway et al. [177] have applied the gray-scale shades to the analysis and validation of cluster structures. Because this example concerns the process analysis that has the

time dimension, the shades of gray were transformed to the time axis for temporal interpretation.

The gray-scaling is actually equal to the zero-one scaling of the data, but it is now applied to the relative locations of the data points in the original space. The gray scale consists of the diagonal values of the well-known RGB-color space[3]. Since the shades of gray are adjusted cluster-wise, the cluster-wise measures are not comparable to each other.

This visual gray-scale approach is not robust, but it is still very informative. The lack of robustness means that only one extreme outlier can make the rest of the points to be visualized almost black. This can be avoided by applying distance ranking (i.e. ordered statistics) and evenly spaced shades of gray, no matter at what distance to the closest prototype data points are lying. On the other hand, outliers can be more effectively recognized when the gray-scale preserves the distance information of the points to the prototype. A large amount of dark shades means that there are some extreme values.

Let us now examine more closely the internal variability of the clusters. The distances for the case $K = 2$ are presented in Figure 57. One easily perceives

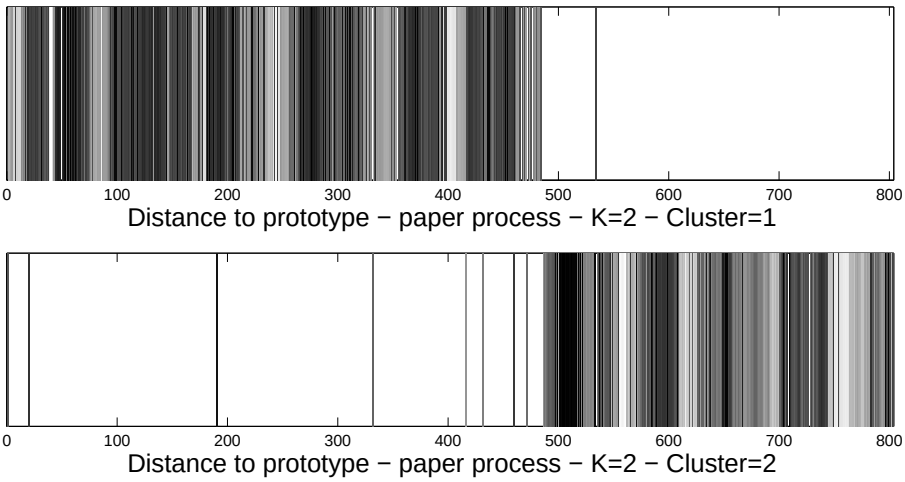


FIGURE 57 Distances to the cluster prototypes when $K = 2$.

that the data structure may contain more than two reasonable clusters or, in other words, process states. This conforms with the outcome of the validity indices. Also by directing the attention to Figure 55 and the classical covariance and SCM based principal component data projections, one can see that the data is fragmented into more than two clusters. Here, one can also see that the prototype-based projection technique, that means the TCM based principal components of the prototypes, loses some information about the local structure of the data, because the cluster-wise data tend to clump very tightly around the cluster prototypes.

[3] RGB is an abbreviation for red, green, and blue color model. RGB is an widely used color model for computer graphics. The other colors are combinations of RGB color intensities (see, e.g., [367])

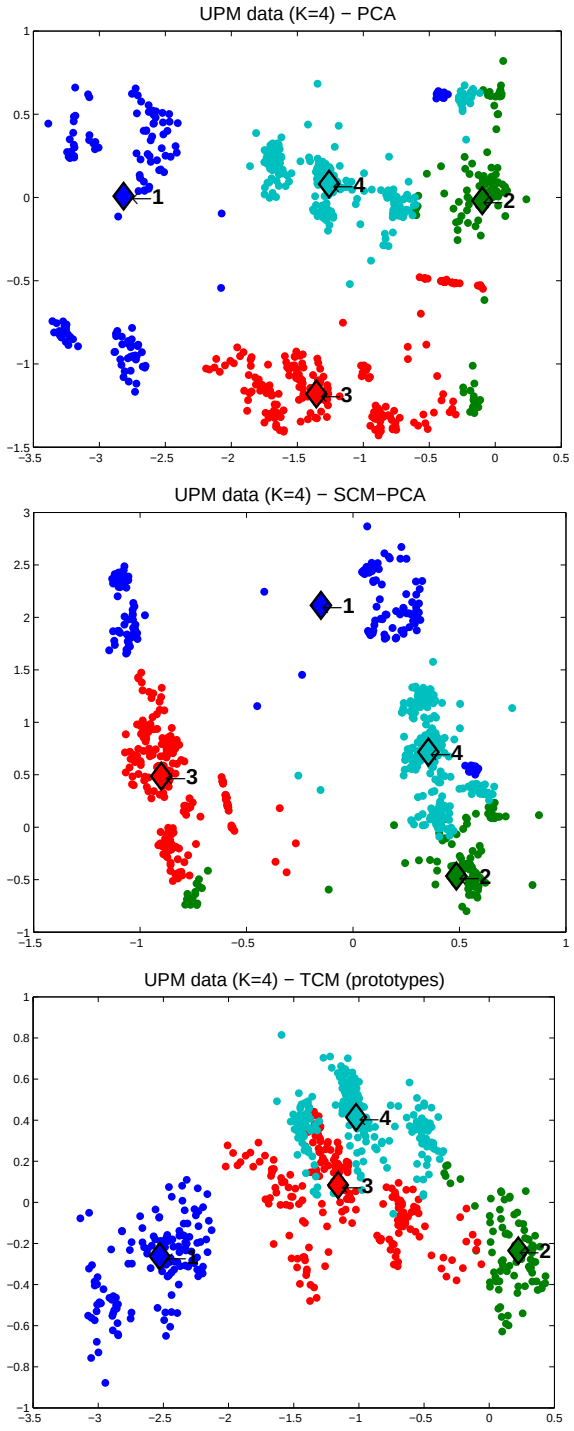


FIGURE 58 The classical, SCM and TCM based principal component projections for 'paper data' in case $K = 4$.

Figure 58 shows the projected two dimensional view to the paper data that is now partitioned into four clusters. The four cluster model is suggested by the ReD validation index as the most inherent partition for the data set (see, Figure 54). However, the projected views presented in Figure 58 do not necessarily support this model. Especially, the cluster number one is quite broadly scattered in

each plot. If we look at the temporal presentations given in Figures 56 and 59, clusters one and two represent short deviating periods in the middle of the two main states of the process.

Correspondingly to the principal components also the cluster-wise distances in Figure 59 suggest that the clusters one and two should both be further partitioned into the sets that occur during the first and second halves of the process. By inspecting the clusters one and two in Figure 59, one can clearly see that the short deviating periods during the latter part of the process are distant to the deviating ones during the first part of the process. Hence, both the temporal and two dimensional views to the data make this four clusters instance of partitioning questionable, even though the ReD validation index supports it.

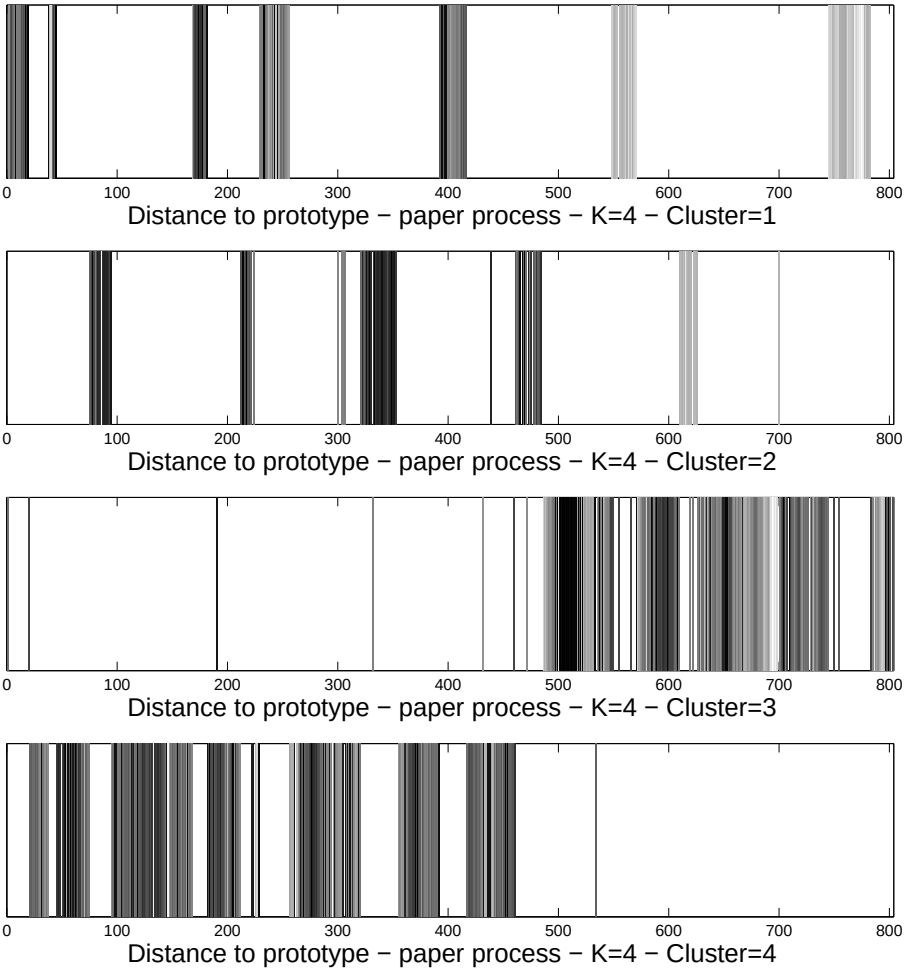


FIGURE 59 Distances to the cluster prototypes when $K = 4$.

As a last case, the clustering solution with $K = 6$ is discussed. According to the new robust silhouette index this is the most inherent partitioning for paper data. By comparing the projected clusters in Figure 60 to the previous cases, it turns out that these are actually hierarchical "sub-clusters" among the clusters obtained using $K = 2$. Moreover, the cluster number one, which was quite fragmented in the four clusters case, is now further partitioned into two sub-clusters,

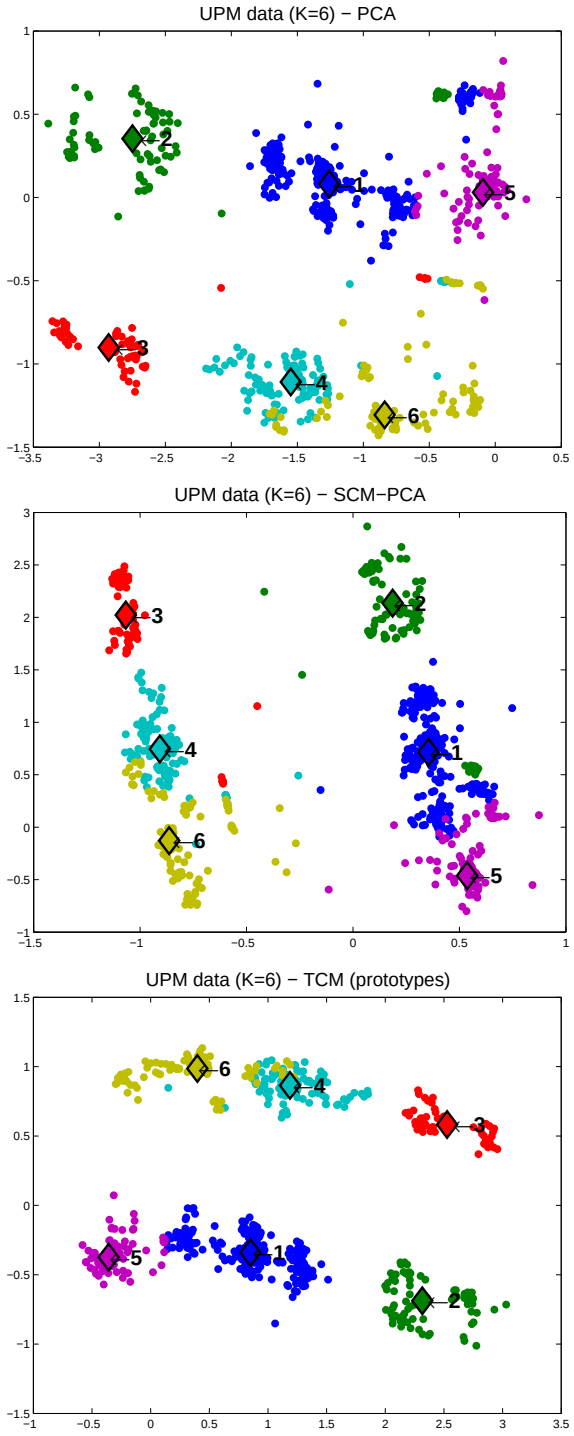


FIGURE 60 The classical, SCM, and TCM based principal component projections for 'paper data' in case $K = 6$.

which leads to better partitioning of the data. The clusters that overlap for $K = 6$ are possibly very close to each other.

All projections in Figure 60 indicate that the cluster number one and five are very close to each other. This seems reasonable as the clusters are, moreover, connected to each other in the temporal sense and some points in the fifth cluster

are very distant to the prototype (see Figures 56 and 61). The fourth and sixth cluster are also overlapped (see, Figure 60). This is also visible in the TCM based principal component projection. On the time axis, the sixth cluster occurs in the middle of the fourth cluster, which also suggests that the assumption about the closeness might be true (see Figure 56). Finally, Figure 56 shows that the both major clusters of the case $K = 2$ are now also temporally further partitioned into three clusters.

The "hierarchical" behavior indicates that the initialization is very stable. It provides consistent solutions for different choices of K, following the hierarchical structure of the data, which sometimes makes it difficult to decide how many clusters there truly exist (cf. discussion about the ambiguity in clusters, Chapter 3). The results show that some intelligence of the hierarchical clustering methods is captured in the new clustering method.

As a result, we have found that the given process period consists of the two major states that are temporally almost unbroken. However, because the clusters are not very compact, which means that the data is possibly compressed too much, the six cluster model seems to be the next appropriate solution for more detailed clustering. The result is supported by the robust silhouette index and ensured by the visual explorations from the three different two dimensional projections, and also by the temporal view to the progress of the process.

For a comparison, the projected clusters for cases $K = 3$, $K = 5$, and $K = 6$ are presented in Appendix 4. The choice $K = 3$ does not provide coherent and compact clusters, since all the clusters seem to be quite broadly scattered. With $K = 5$, the fifth cluster is still very non-uniform whereas the case $K = 7$ leads to small clusters that are not well-separated anymore. Thereby, $K = 6$ seems to be a very reasonable guess for the number of clusters in the paper data case.

8.3.2 Image quantization

As an image is a kind of multivariate vector signal, clustering of its colors is often referred to as vector or signal quantization, because it leads to a reduced number of colors and image data. Thus clustering of image colors is also called simply image compression. Quantization of images [140, 175] can be used as a preprocessing method, for example, in context of image database mining.

In Chapter 2, a couple of applications from the fields of astronomy and geoscience are presented. Furthermore, modern surveillance and reconnaissance applications transmit and process a lot of image data under strict real-time requirements and limited availability of resources [123]. The image quantization may be of use for such applications, since the reduced size and simplified color structure make data transmission, knowledge mining, and object searching from large image databases easier as long as the most significant information remains in the image.

As images are composed of large amounts of data they also provide a good sample application for the DM purposes. For this problem, the robust clustering method with the validation index is applied. The unnecessary shades of colors

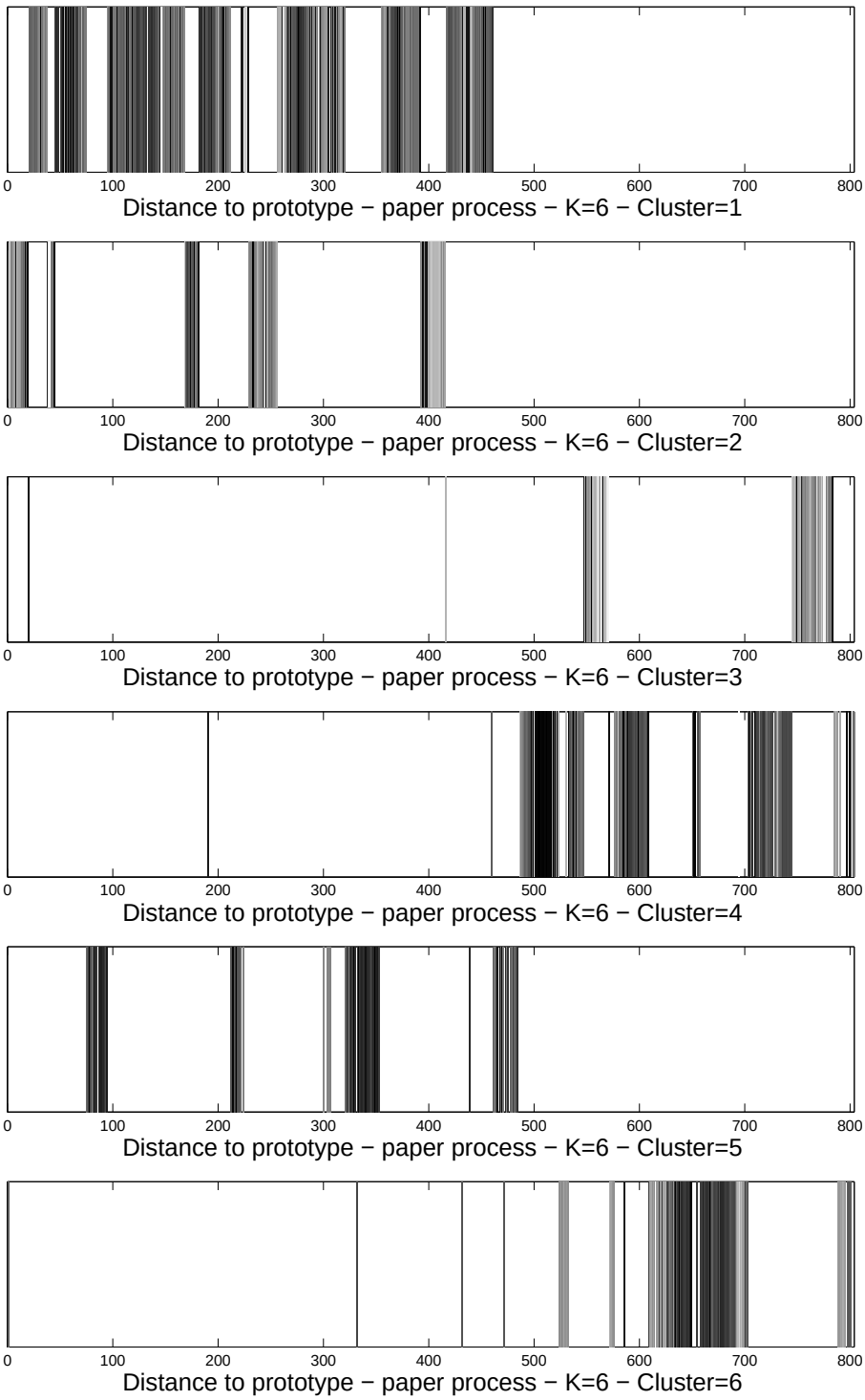


FIGURE 61 Distances to the cluster prototypes when $K = 6$.

should be compressed away and replaced by ”prototype colors”. The results are evaluated by comparing how well the available information in the original figure is kept in the quantized figure. The heuristic validity indices are used to predict the sufficient number of colors. It is desirable that the indices estimate the number of necessary colors to present the major objects in the figure. The goodness of

solutions can be evaluated simply by the eye from the resulting images.

Standard test images

FIGURE 62 Standard test images. On the top row from left house and peppers images. On the bottom row an image of Lena.

As examples of image compression applications, three standard test images, shown in Figure 62, were used. All images are 3×8 bit true color RGB images. Each component of RGB is treated as a variable, which means that $p = 3$. The images do not contain erroneous or missing data.

The house image consists of 256×256 RGB pixels. As the image pixels are three dimensional RGB color values, the size of the house image data matrix is 65536×3.

The peppers image is an RGB image of size $200 \times 200 \times 3$. This is reduced from the original peppers image, available on the Web, and consisting of 512 pixels. The size is compressed in order to reduce the time that is required for computation.

The Lena image is an RGB image of size $512 \times 512 \times 3$. Due to the large size of the problem, the estimate for the correct number of clusters is computed only by the robust silhouette heuristics in the case of Lena image.

Results

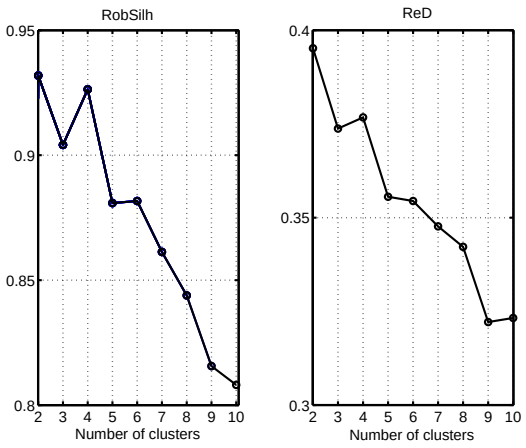


FIGURE 63 Indices for the number of clusters in the house image.

FIGURE 64 Clustered house image for $K = 2$.

To start with, let us consider the house image for which the both of the heuristics suggested $K = 2$ for the number of clusters. This means that one bit (black and white) is enough to approximate the patterns in the image without color information. The index curves are shown in Figure 63. The curve goes down almost constantly from $K = 2$ except for $K = 4$ and $K = 6$ where some deviations exist. The compressed image of the case $K = 2$ is presented in Figure 64.

When compared to the original image, one can see that many constituent parts of the image remain recognizable. It reveals the house with different elements, such as windows, roof, guttering, waterspout, fascias, and chimney. Also the cloudless background is observable.

The lost information are the bricks on the wall and the shadows. This kind of information could be useful, for example, when one needs to individualize the house from the image or know the time of the day when the picture was taken.

However, much significant information is preserved in the image even though the size is compressed from $256 \times 256 \times 24$ bits to $256 \times 256 \times 1$ bits, that is to $4,7\%$ of the original image. If the color information is added, additional 24 bits RGB words are needed that can be then pointed by the bitwise pointer.

Let us consider a more colorful image with a number of objects and shadows next. The peppers image in Figure 62 presents a number of peppers with

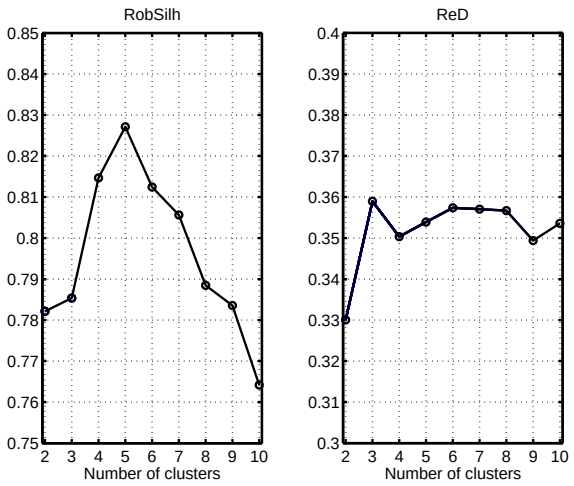


FIGURE 65 Indices for the number of clusters in peppers image.

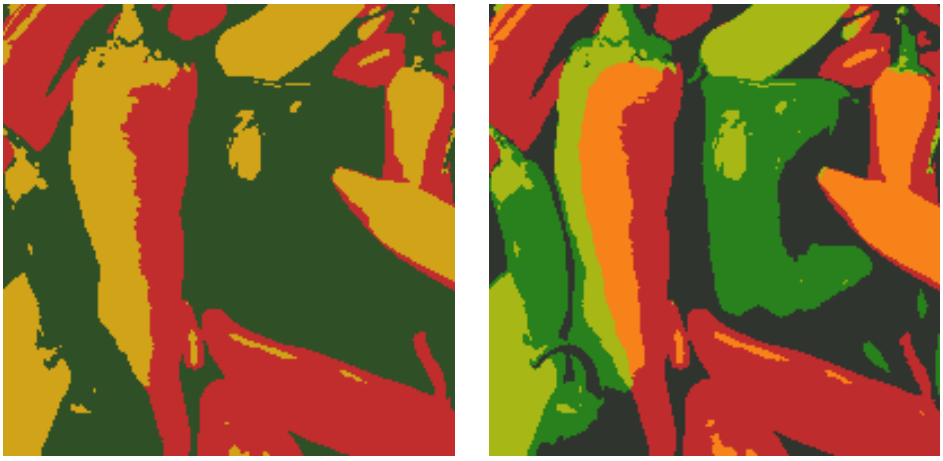


FIGURE 66 Clustered peppers for $K = 3$ (left) and $K = 5$ (right).

different sizes and colors. Furthermore, there are shadows and reflections that expose the direction of the arriving rays of lights. The two indices propose different K's for this image. The ReD index suggests that the inherent number of colors is $K = 3$, whereas the robust silhouette index yields $K = 5$ (see Figure 65). When looking at the uncompressed image in Figure 62, one can easily count four significant colors: red, orange, green, and yellow. Moreover, dark shadows and light reflections represent important shades of the colors. In this case they are actually very close to black and white.

When comparing the compressed images in Figure 66, one perceives that three colors is not enough to show all the information visible in the original peppers image. The information loss is, for instance, counted in the number of peppers that have been faded away. The direction of arrival of the light rays is still visible, but the green peppers are shaded to the dark shadow.

When looking at the right image of the same figure, one can see that five colors is enough to separate the peppers. The yellow peppers are shaded with the color of the light reflections on the green peppers and the orange peppers with the color of the reflections on the red peppers. The dark shade represents the shadows.

Hence, $K = 5$ clustering yields very reasonable quantization for the peppers image and preserves the significant information. In other words, the robust silhouette index is able to point out the most significant colors from the image. It finds the colors that exist in large amounts with slightly different shades and discards the rest.

As a result, one can compress the peppers image data by defining pixel-wise pointers to a five color RGB map. The five color map requires three bit pointer. Hence, the significant information expressed by $200 \times 200 \times 3 \times 8$ bits can be compressed by the pointer technique into $200 \times 200 \times 3$ bits plus the 5×24 bits color map, which is approximately $12,5\%$ of the original amount of data.

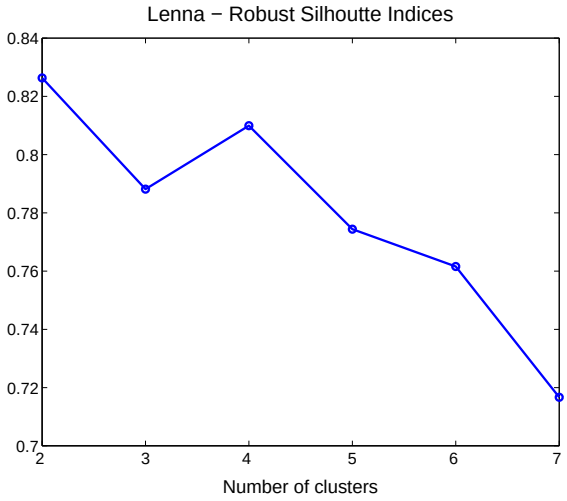


FIGURE 67 Robust silhouette index for $K = 2, \ldots, 7$ in Lena image.

The last example, the well-known Lena image, is the most difficult to interpret. The focus is to find out how the change in the robust silhouette index reflects the change in the quality of the image. Hence, the Lena image is clustered for $K = 1, \ldots, 7$ and the robust silhouette indices are computed for each K. The index proposes that the most significant information is compressed by only two colors (see Figure 67).

The second best index is obtained for $K = 4$. Actually the most significant information of the Lena image is observable with only two colors (see Figure 68). One can observe the figure of the woman from the image, but the shape of

FIGURE 68 Clustered Lena images for $K = 2, \ldots, 7$ (left to right and top down).

the nose and lips have been faded out by the compression. The shadows have also disappeared. With $K = 4$ obvious enhancements to the image quality are obtained as the features and shadows have become clearly visible.

Increasing the number of clusters from $K = 2$ to $K = 3$ or from $K = 4$ to $K = 5$ does not produce significant changes to the image quality when compared to enhancement between the three and four cluster images. This observation correlates to the shape of the robust silhouette index curve. The increment of K

without significant gains of image quality should reduce the value of a reasonable index.

Discussion

The image data has been used as an example of data clustering and cluster validation. The results show clearly that the cluster method used is able to recognize the most important colors in the particular cases. Thereby the objects or patterns can be recognized from the images with less colors. This may provide advancements, for example, for edge detection, segmentation, pattern recognition, and image database mining applications. In edge detection, one tries to find the points of the image where luminous intensity changes sharply (e.g., [146]). By finding the edges of the image objects, one can determine the shape of these objects and recognize them better and quickly. This helps, in turn, in finding the interesting objects from large image databases. As image data contain huge numbers of pixels, filtering techniques can be used to reduce the amount of computation (see, e.g., [175, p.467]). To this end, in these experiments, the computing time was relatively long (hours), especially for the indices.

8.3.3 Software project data

In this example, a data set consisting of 3024 software projects is clustered. The data was collected by ISBSG[4] and is available on a CD-rom [202]. It is intended to benchmark, evaluation, and estimation tasks for software projects and organizations. Moreover, as in this thesis, it can be used for software engineering research. The data represent projects from 20 different countries. The software products of the projects represent several applications and systems, including management information, transaction/production, process control, and mission critical real-time systems among others. Several different process model, programming languages, and platforms are also covered.

The data attributes provide information about several factors such as sizing, effort, productivity, schedule, quality, and so on. Although there is a lot of background knowledge about the projects available on the CD-ROM, the exploitation would require heavy data type conversions and preprocessing operations due to the problematic data presentation principles. Thus, it is not very much utilized in these experiments. The data contain also huge amounts of missing values.

The clustering result is compared to the domain knowledge about the quality of the project-wise data. Data quality rating (DQR) describes the quality and integrity of the project data submitted to the collection from different projects and organizations. It is a kind of measure on credibility of the submitted project data sets that classifies the projects into four classes (A, B, C, and D). Unadjusted Function Point Rating (UFPR) measures the credibility of the data provided about the functional size. Four classes are used also for this information (A, B, C, and D). In both cases the class A signifies the highest credibility.

[4] The International Software Benchmarking Standards Group (http://www.isbsg.org/)

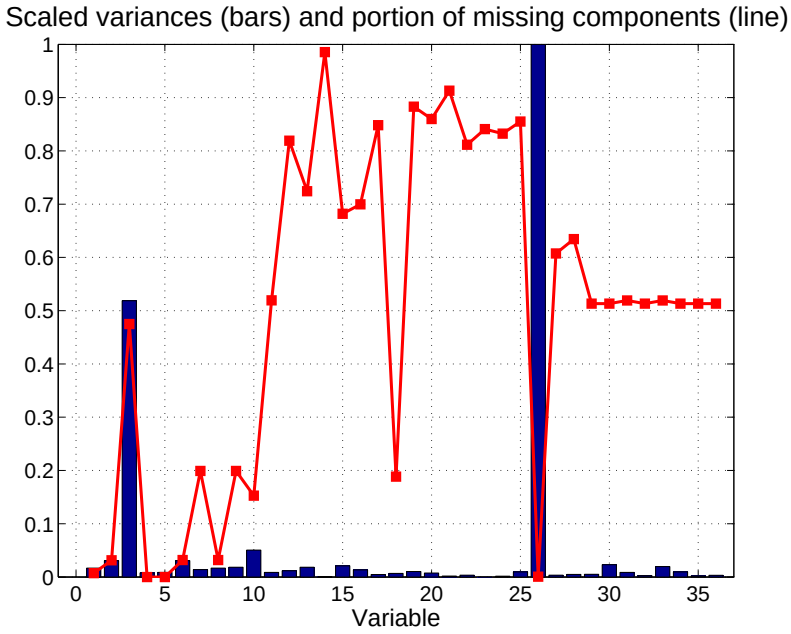


FIGURE 69 Scaled variable-wise variances (blue bars) and the proportion of missing data (red line).

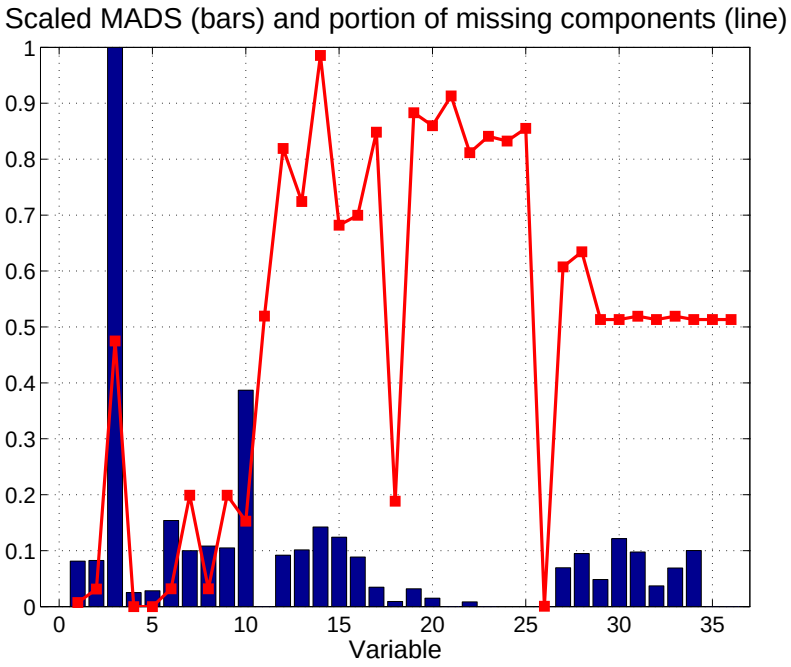


FIGURE 70 Scaled variable-wise MADs (blue bars) and the proportion of missing data (red line).

In order to perform the experimental cluster analysis for the data set, 36 numerical attributes were chosen from the data set (The list of the attributes used is given in Appendix 5). The fraction of missing values remaining in the selected data is 49,9%, which means that it is extremely sparse. The variable-wise fractions of missing data are depicted in both Figures 69 and 70. One can see that for

many attributes, more than half of the data are missing. The time needed for the overall computation is some minutes.

The relative variances and MADs are given in the same figures. The bar plots show that there are very large differences in the relative dispersions of the variables. At the same time, there are only a few variables that have considerable variability. The largest variance is found for variable numbers 3 (Value Adjustment Factor) and 26 (Resource Level). On the other hand, the more robust MAD dispersion estimate shows high variability solely for the variable number 3. This is due to the fact that more than half of the observations for the discrete variable number 26 accumulate to a single value that is one in this case.

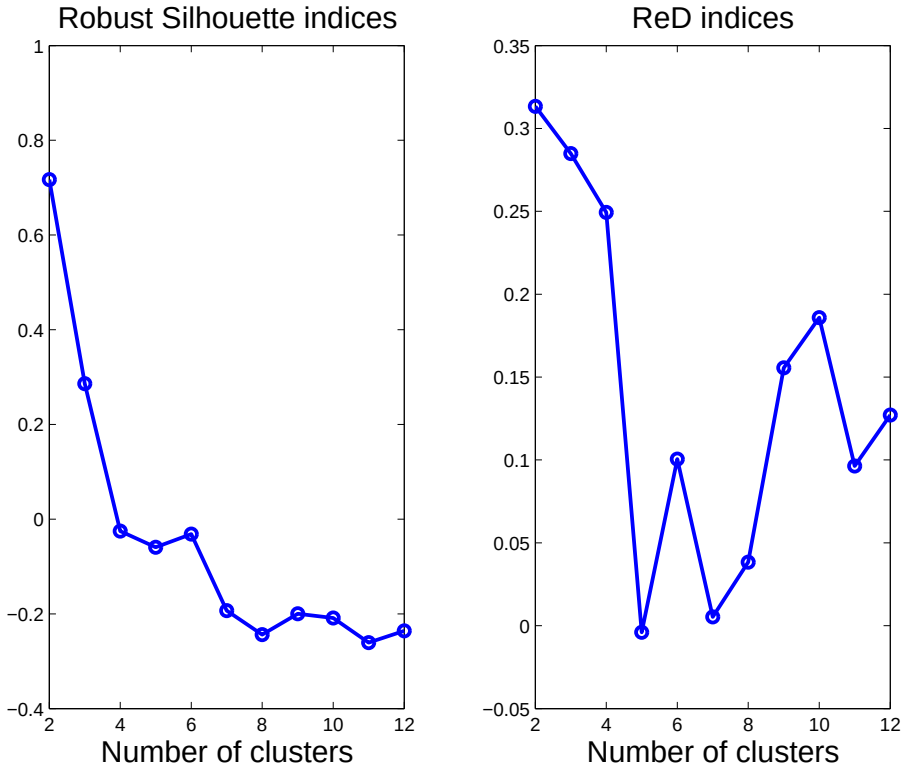


FIGURE 71 Robust silhouettes and ReD indices for $K = 2, \ldots, 12$.

Figure 71 shows the values of the robust silhouette and ReD indices for $K = 2, \ldots, 12$ after the clustering. The both indices propose that the case $K = 2$ fits the best to the data. Two dimensional projections of the result are shown in Figure 73. One can see that the classical principal components break down and the data becomes very flat in the direction of the second principal component. Also TCM based principal components lead to shrunken data projections. SCM based principal components retain the spread of the data set to some extent.

The breakdown of the classical variance based principal components indicates that the data contains extreme values in the direction of the largest variance. The cluster prototypes are clearly emphasized at the direction of one variable. This leads to the flat data clusters in the TCM based projection of the two cluster prototypes. The cluster centers are so distant to each other that the within-cluster

variation almost vanishes from the clusters. A closer look into the variable-wise

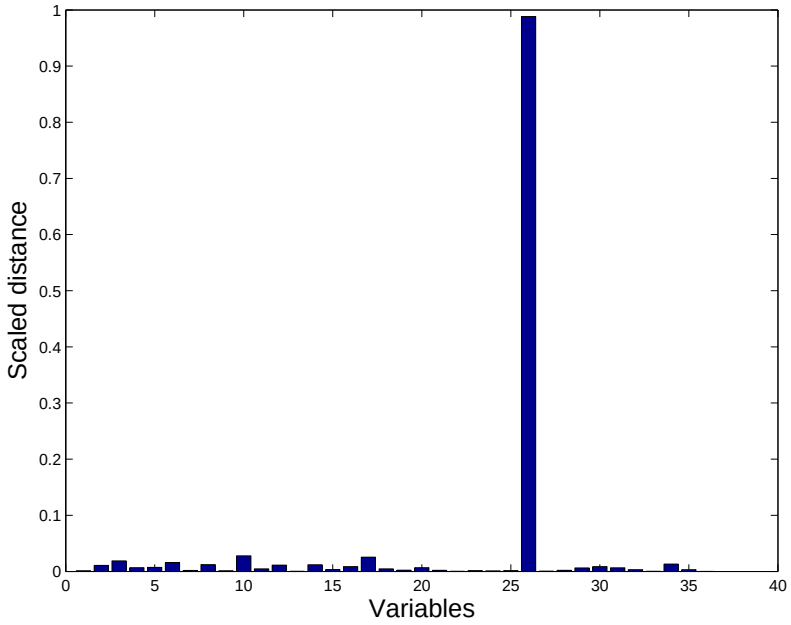


FIGURE 72 Scaled variable-wise distances between prototypes.

distances between the cluster prototypes shows that one variable dominates the separation. In Figure 72, the between-clusters distance in the directions of all variables are normalized by the range of the variables. One can see that variable 26 (Resource level) is the most discriminative feature as it attracts the prototypes to the both ends of the variable range. Resource level is a categorical variable whose permitted values are $\{1, 2, 3, 4\}$.

A closer look into the partitioning shows that the cluster number one has captured all the projects for which the resource level is 3 or 4. Correspondingly, the cluster two has captured the projects for which the resource level is 1 or 2. The mid-point of this range partly divides the data into two clusters. The cluster prototypes are, moreover, located very near to the outermost values of the resource level, because the data points in both clusters emphasize the outermost alternatives of the resource level variable (Cluster 1: 22 points on resource level 3 and 341 on resource level 4. Cluster 2: 2412 points on resource level 1 and 247 points on resource level 2.)

The SCM based projection gives a more dispersed view to the data and clustering (see the middle plot in Figure 73). The cluster structure does not seem very coherent in this view. Using this picture one might predict six clusters partitioning for the data. The six cluster case also indicates slightly deviating change to the shape of the decreasing curve of the validation indices. The clusters of the six cluster case are presented in Figure 74. One can see that the resulting clusters are not very compact. The rest of the projected views to the clustering solutions are presented in Appendix 6.

To give a domain knowledge point of view, the clustering solutions are compared to the classification of the data rating indices DQR and UFPR. Because the

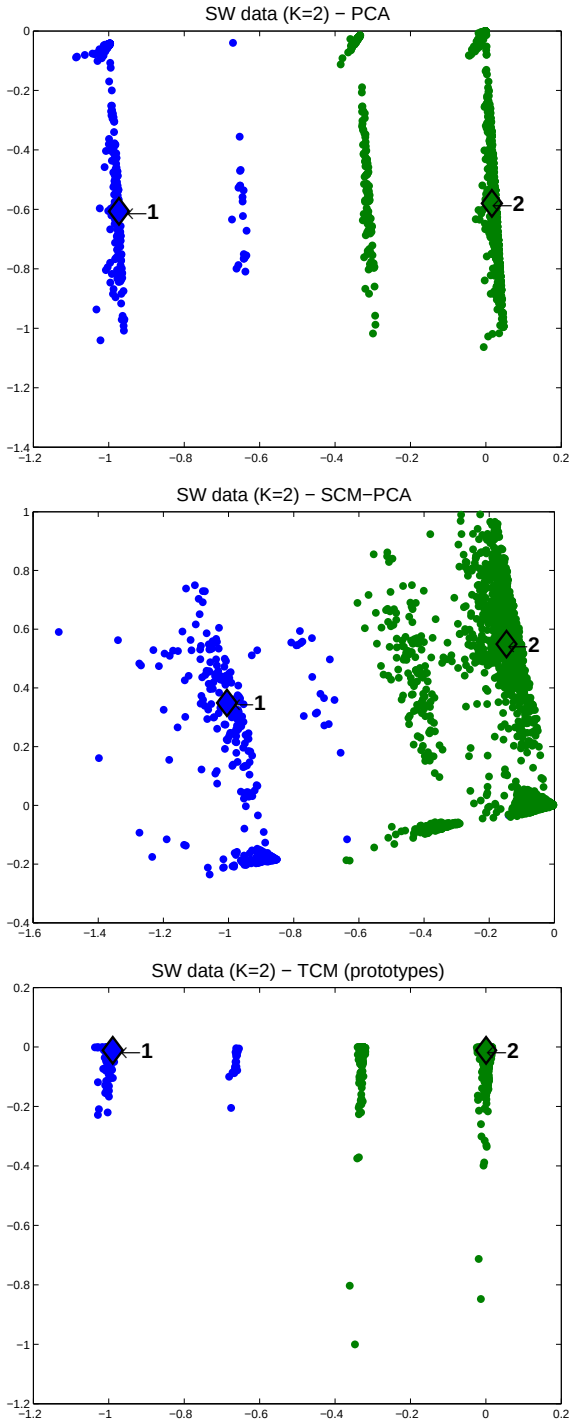


FIGURE 73 The classical, SCM, and TCM based principal component projections for the project data in the case $K = 2$.

clusters do not match with the rating classes, it is clear that the cluster-wise distributions of the given DQR and UFPR values do not follow the obtained cluster structures.

Tables 11-14 show the cluster-wise distributions of the class values. Clearly, there are no clear relationships between the cluster structures and credibility ratings. This is actually a desirable result, since otherwise the knowledge about the

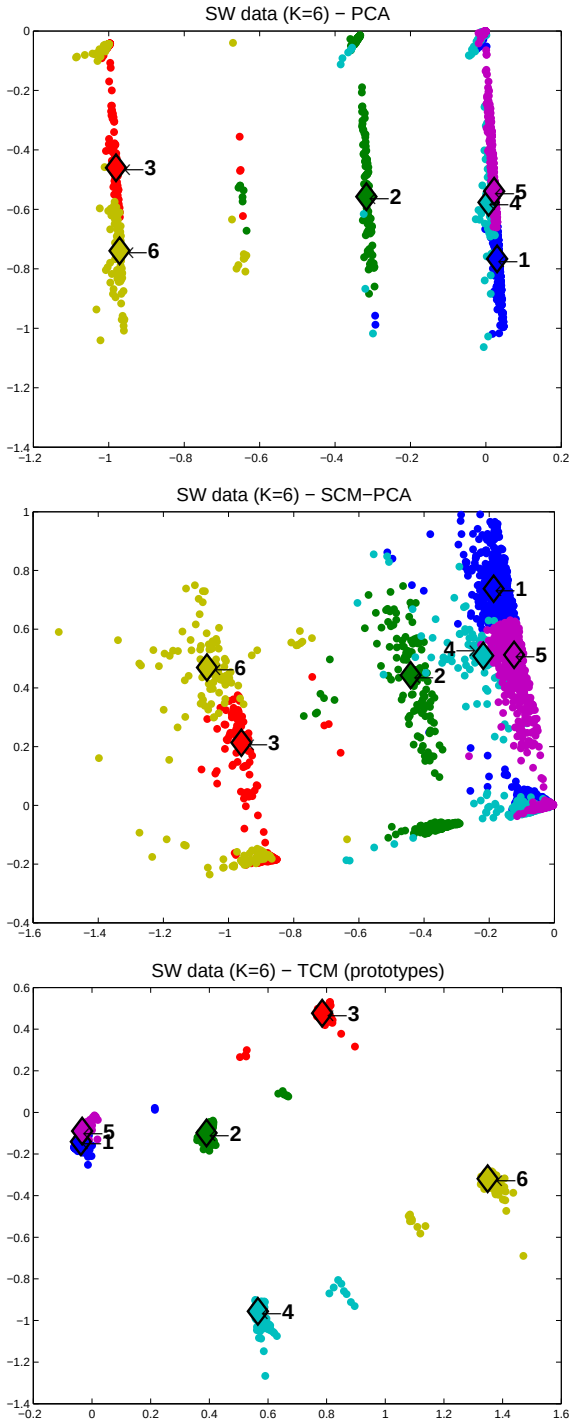


FIGURE 74 The classical, SCM, and TCM based principal component projections for the project data in the case $K = 6$.

software projects, that is represented by the numbers, could be biased towards the quality issues of the data collection strategy.

To summarize, the software project data is very difficult for the clustering algorithms as the data tend to accumulate to the low numbers of the variable-wise data range. Actually, in 34 out of the 36 chosen variables data almost correlate to each other. The attribute number 3, namely the value adjustment factor, is the

TABLE 11 Division of DQR classes (A,B,C,D) into clusters when K=2.

	A	B	C	D
Cluster 1	10.35	12.11	17.97	15.09
Cluster 2	89.65	87.89	82.03	84.91

TABLE 12 Division of UFPR classes (A,B,C,D) into clusters when K=2.

	A	B	C	D	Empty
Cluster 1	8.53	0	19.28	33.33	52.63
Cluster 2	91.47	100.00	80.72	66.67	47.37

TABLE 13 Division of DQR classes (A,B,C,D) into clusters when K=4.

	A	B	C	D
Cluster 1	6.27	6.61	3.91	10.38
Cluster 2	24.39	40.13	32.81	24.53
Cluster 3	65.26	47.76	49.22	60.38
Cluster 4	4.09	5.50	14.06	4.72

TABLE 14 Division of UFPR classes (A,B,C,D) into clusters when K=4.

	A	B	C	D	Empty
Cluster 1	3.94	0	9.90	33.33	47.37
Cluster 2	34.91	40.91	38.91	25.00	27.37
Cluster 3	56.56	59.09	41.81	41.67	20.00
Cluster 4	4.59	0	9.39	0	5.26

only variable with a somewhat symmetric data distribution. After the scaling to the range $[0, 1]$, most of the variable-wise distributions resemble very much the shape of the exponential distribution (see Figure 75). Moreover the tails of the distributions are heavy suggesting several outliers. Hence, the correct answer to the clustering problem might also be that the project data consists of a single non-symmetric and skewed cluster.

8.4 Discussion

In this chapter, three clustering examples using real-world data sets were given. At first, a data set from process industry was clustered and the results were visualized in many ways. The new validation index referred to as robust silhouettes indicated reasonable numbers of clusters. The experiment shows that one can reach, in a few minutes, an extensive understanding about industrial processes by clustering and by reasonable visualization techniques. This can greatly assist the domain specialists in their work to control and enhance the process by integrating most of the available data and knowledge.

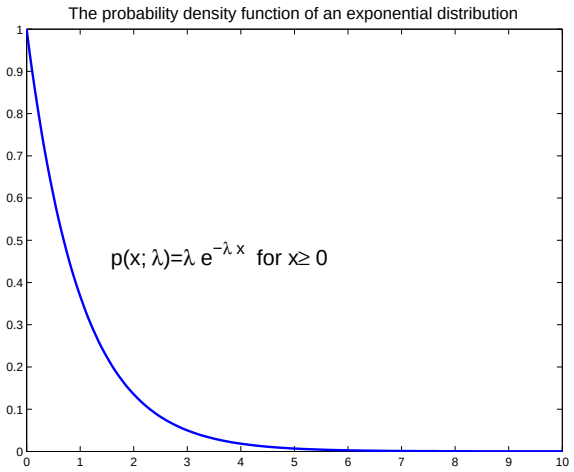


FIGURE 75 The shape of the pdf of the exponential distributions.

In the second experiment, a set of RGB-color images were clustered to find the essential colors and, thereby, compress the data size. Image data are of huge size in the number of pixels and, therefore, heavy load to any computational processing. The clustering and the robust silhouette index showed reasonable performance in this case as well.

The last one, that is the software project data set, contained the most troublesome internal structure from the data clustering point of view. The method ended up with the two cluster solution, which was the smallest number that was tried. Because the data contains more than 50% of contamination, it is not surprising that even robust clustering methods fail in finding reasonable groups from it. Closer investigation of the data set indicated that perhaps the data is best described by only one special shaped density function rather than several clusters.

Robust covariance matrix estimates were also used in data projection and visualization. SCM without robust scaling in the directions of principal components retained better the variability than the classical PCA projection. As the robust methods are computationally intensive, a TCM covariance matrix with robust scale estimates to the principal components was applied to the cluster prototypes in order to save time and obtain cluster-structure preserving projections (cf. [86]). It seems that cluster based projections ignore local inner-clusters features from the data.

To summarize, the methods seem to efficiently uncover the essential features from various data sets. The efficient and accurate algorithms enable building the clustering methods using robust techniques. By applying the available case strategy to missing data in all computation, minimal data preprocessing is required on real-world data sets. A number of interesting large-scale applications still remain as a future challenge, such as, e.g., software quality analysis using code metrics [266], and biomedical applications [1] among many others.

9 CONCLUSIONS

Data mining and knowledge discovery is a modern and multidisciplinary field of science and business. Thereby, it provides a number of challenges for a variety of experts. The problems originate from a diverse set of companies, organizations, and institutions, where large amounts of data are constantly used, processed, and managed. Earlier, commonly used database queries were enough to handle all available data, because the domain experts knew the facts they needed to find. In the present situation, such content-based methods, which are actually based on the assumption that the user knows exactly what she/he is searching for, are not enough. If one wants to keep up with others, especially concerning the field of rapidly globalizing business world, she/he should also be interested in finding unexpected facts. This information provides, perhaps, the most critical form of competitive intelligence in the field of today's business world. The discoverer of an underlying, unexpected, and useful fact is one step ahead of the competitors.

On the other hand, ROI can not always be measured in money. Many fields of science, such as medicine, yield a lot of "human value" through better understanding of the collected data sets. The results may be realized in the form of healthy diets, efficient medicines, awareness of carcinogenic substances, etc., that finally lead to a healthier and longer life. Moreover, the enhanced control and smart decision making concerning large-scale industrial processes can be obtained by uncovering and understanding the most essential facts residing in those hundreds and thousands of measurements taken from the processes. This may not only increase the owner's income, but may also lead to remarkable energy and raw material savings that are significant gains from the ecological perspective.

Thus, the topic of this thesis provides a lot of value for many directions. As the contents of this thesis show, to be fully utilized, further development and practical application of KM require a lot of co-operation among the experts from related fields. An applied KM research project may include experts ranging from researchers to domain experts and from mathematicians to information systems and organization specialists.

The main results of this thesis are summarized next.

- Based on the ideas and practical experience from applied industrial processes, a new merged process model, knowledge mining (KM), was proposed. The model separates domain and method expertise into the KD and DM processes, respectively. In this model, a domain expert is able to accomplish knowledge discovery tasks without concern about complex computational and statistical details. To assist domain experts in managing and finding useful data, the idea of using the genre-based analysis method was proposed. On the other hand, due to the clear interfaces between the processes, DM experts who understand methods of data processing and model building, can be exempted from the domain level issues.

- As the major result of the thesis, a new robust clustering method with several desirable properties was developed. The development of the method was preceded by an extensive survey on the concepts and elements of data clustering. The survey clarified several complex issues, such as various definitions of the concept of cluster and ambiguous nature of the "correct" number of clusters. These are likely met by all method developers. These problems also explained the existence of so many clustering methods. Later, the results of the survey were further supported by the results of the numerical experiments. The new and robust clustering method involves several desirable properties from the KM point of view:

  - It is highly automated (i.e., minimal amount of user inputs are required).
  - The method inherently and efficiently handles missing values and tolerates large amounts of erroneous data, which frees the user from most of the difficult preprocessing problems, such as outlier pruning and imputation.
  - It produces more consistent results than the variants that are based on the normal assumptions.
  - The use of the proposed robust silhouette index seems to yield reasonable results on the best number of clusters on real-data experiments.
  - The real-world experiments show that the robust clustering yields useful results for mutually very different real-life applications.

- On the side of the clustering method development, new formulations, algorithms, and proofs were produced for the non-smooth problem of the spatial median. The numerical and statistical experiments with extensive discussions showed the accuracy, scalability, and efficiency of the proposed SOR-algorithm. One should also note that the problem has also been widely considered in the fields of location/management science and operation research, which means that the results have also multidisciplinary value (e.g. [262]). Thorough exploration of the computational and statistical properties of the l_1- and l_2-based multivariate M-estimators produced a lot of useful

information for further development of the method. The somewhat shallow analytical proofs concerning the spatial median problem is perhaps the weakness of this thesis (at least from the point of view of mathematical analysis, since the proofs do not extend to the missing data treatment and the convergence of the acceleration step of the SOR-type spatial median solver was not shown). However, this part of the thesis has already been followed in a more involved mathematical analysis by Valkonen [381].

- Robust covariance estimates were introduced for dimensionality reduction and data visualization purposes. The experiments showed that these are valuable tools for data mining applications. The class structure was better retained in the projected views when the robust covariance estimates were used. The available data strategy was also applied to the missing data.

- The experiments on real-life data showed the usefulness of data clustering for analysis and investigations of industrial processes and image segmentation. The essential information of the test images were quantized into a couple of colors. This result is of great value when mining large image databases. The fluctuation of an industrial process was well characterized by the obtained cluster structures. The proposed graphical techniques clearly assisted in the interpretation of the results. These results were also successfully validated together with domain experts. On the other hand, results on the software project data showed that data clustering was not the best approach to knowledge mining in this case due to the inherent lack of multivariate structure.

Overall, robustness seems to be a gainful property for data clustering, not only when there is contamination in the data, but also in the presence of normal conditions. Robust clustering seems to produce more consistent results, even if the data were normally distributed. Moreover, due to the smaller number of algorithm iterations, the computing time was also shortened when compared to classical methods. However, the numerical experiments of the initialization methods supported the fact that universal clustering methods are impossible to define. Although the robust robBF method was on an average considered the most useable method, KKZ methods were clearly the best performers on well-defined clusters. Trimming seemed to be useful technique in certain cases, but more practical tests are still needed before more general usage. A problem is, of course, the trimming fraction parameter that is difficult to define and in conflict with the black-box principle.

The strong side of this thesis is its comprehensive scope that extends from analytical proofs to business processes. In the end, this is really what knowledge mining (or data mining and knowledge discovery) is about. The available information about the properties of the used methods and estimates were also strictly explored and discussed. The thorough numerical and statistical testing also ascertained and uncovered several interesting and useful facts about the methods. The rigorous analysis of the numerical results have clearly shown that robustness does not only provide advantage on erroneous data, but may lead to correct and fast results also on well-defined cases.

9.1 Future work

Finally, a few words about future challenges that have emerged during the course of this work. The proposed robust clustering method provides some immediate possibilities for further experiments. The robust refinement initialization method, robBF, produces information about stability or uncertainty of the obtained clustering that can be exploited for estimating the number of clusters (cf. resampling-based heuristics [250, 336, 95, 243]). One could, for instance, measure the variation in the locations of initial cluster prototypes that are computed on the chosen number of sub-datasets. Large variation in the locations of sub-dataset cluster prototypes indicates uncertainty in the number of clusters. This approach was not used in the experiments, but it is an open possibility for development.

Another open question is a reasonable scaling of variables. Especially binary variables were experienced problematic as they possess maximum variability after scaling the data to the interval from zero to one. This gives highest possible discriminative weight for all binary variables as they always obtain values 0 and 1. One could rather consider binary values as representative values 0.25 and 0.75 of the first and second parts of the range, respectively.

In the applied research projects, the results interpretation was assisted by ranking variables and plotting the cluster-wise distributions of the variables. After data clustering, a ranking index that expresses the discriminative power for each variable can be computed. One can take as the ranking index, for example, the correlation between the cluster-wise distributions or ratio of the within and between cluster scatters. As the variables are ordered, one obtains a rapid overview on the distributions of the clusters of the most discriminative variables by inspecting the cluster-wise histograms. The difficulty of this approach is the choice of the best ranking index. However, promising results were obtained in the applied projects by ranking the variables so that the variable with the smallest average correlation among the cluster-wise distributions (i.e., histograms) received the highest ranking.

As illustrated in Section 3.2.1, missing data complicates the comparison of distances between different points. This problem also occurs with the KKZ initialization method in which the distances between data points are compared. A missing value can actually be considered as an extreme outlier (it can have any value). As the data points with missing values on non-overlapping variables are in different subspaces, they are difficult to order. As a future challenge, a subspace-ranking index for data points might be useful. The idea is that the less common the variables in a pair of data points, the more distant the data points are from each other. This approach was not applied in this thesis, but with an efficient implementation this could be worth of some experiments.

As the cluster is a very application and data dependent concept, a single clustering method may not be enough for mining heterogeneous data sets. Perhaps the most stable and reliable results are obtained by using several indices, parameter settings, and clustering algorithms, and then performing ”voting” or

finding the largest agreement between the different results, e.g., [173, 35, 370, 143, 106] (cf. committee machines in neural networks [181]). The survey in Chapter 3 showed that there are many changeable elements in the clustering problem that can be varied in order to build a clustering ensemble. One may use different missing data strategies, norms, trimming fractions, and so on. On the other hand, it should be noted that bad results can be obtained also with this approach when the voting is based on wrong assumptions (normality, missing data, outliers, etc.). Hence, different ensemble clustering methods are of great interest for the DM applications, where typically the characteristics of the target data sets are not known. A probable drawback with ensemble methods might be their computational efficiency, as all the methods should be highly scalable. There are several clustering algorithms whose robustness could be enhanced with minimal changes. One of the most interesting is the LBG-U-method [134]. The idea is to remove the cluster with minimum utility and move the prototype to a new location. The minimum utility is defined as a cluster with the smallest contribution to the value of the clustering criterion function. The prototype of the minimum utility cluster is moved to a cluster that currently generates the largest contribution to the value of the clustering criterion function, which is then split between the old and new prototypes. This approach always decreases the value of the criterion function and terminates in a finite number of steps. The method is not assumed to be as sensitive to initial conditions as, for example, the original K-means clustering methods that is a local-search strategy. One can easily see that the LBG-U-method provides several possibilities in the utilization of robust principles.

REFERENCES

[1] *Ercim news 60 (special: Biomedical informatics)*, January 2005.

[2] B. ABOLHASSANI, J. SALT, AND D. DODDS, *A two-phase genetic k-means algorithm for placement of radioports in cellular networks*, IEEE Transactions on Systems, Man and Cybernetics, Part B, 34 (2004), pp. 533–538.

[3] R. AGRAWAL, J. GEHRKE, D. GUNOPULOS, AND P. RAGHAVAN, *Automatic subspace clustering of high dimensional data for data mining applications*, in Proceedings ACM SIGMOD International Conference on Management of Data, L. M. Haas and A. Tiwary, eds., ACM Press, 1998, pp. 94–105.

[4] R. AGRAWAL, T. IMIELINSKI, AND A. N. SWAMI, *Mining association rules between sets of items in large databases*, in Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, P. Buneman and S. Jajodia, eds., Washington, D.C., 26–28 1993, pp. 207–216.

[5] R. AGRAWAL, H. MANNILA, R. SRIKANT, H. TOIVONEN, AND A. I. VERKAMO, *Fast discovery of association rules*, in Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, 1996, pp. 307–328.

[6] M. B. AL-DAOUD AND S. A. ROBERTS, *New methods for the initialisation of clusters*, Pattern Recognition Letters, 17 (1996), pp. 451–455.

[7] M. S. ALDENDERFER, *Methods of cluster validation for archaeology*, World Archaeology, 14 (1982), pp. 61–72.

[8] M. S. ALDENDERFER AND R. K. BLASHFIELD, *Cluster analysis*, Sage Publications, London, England, 1984.

[9] M. R. ANDERBERG, *Cluster analysis for applications*, Academic Press, Inc., London, 1973.

[10] E. ANDRASSYOVA AND J. PARALIC, *Knowledge discovery in databases, a comparison of different views*, in Proceedings of the 10th International Conference on Information and Intelligent Systems - IIS'99 (on CD), Varazdin, Croatia, September 1999.

[11] L. ÁNGEL GARCÍA-ESCUDERO AND A. GORDALIZA, *Robustness properties of k means and trimmed k means*, Journal of the American Statistical Association, 94 (1999), pp. 956–969.

[12] M. ANKERST, M. BREUNIG, H.-P. KRIEGEL, AND J. SANDER, *Optics: Ordering points to identify the clustering structure*, in Proceedings of ACM-SIGMOD Conference on Management of Data, 1999, pp. 49–60.

[13] S. ANSARI, R. KOHAVI, L. MASON, AND Z. ZHENG, *Integrating e-commerce and data mining: Architecture and challenges*, in ICDM '01: Proceedings of

the 2001 IEEE International Conference on Data Mining, Washington, DC, USA, 2001, IEEE Computer Society, pp. 27–34.

[14] C. APTE, B. LIU, E. P. D. PEDNAULT, AND P. SMYTH, *Business applications of data mining*, Communications of the ACM, 45 (2002), pp. 49–53.

[15] F. BACHMANN AND L. BASS, *Managing variability in software architectures*, in SSR '01: Proceedings of the 2001 symposium on Software reusability, New York, USA, 2001, ACM Press, pp. 126–132.

[16] K. D. BAILEY, *Cluster analysis*, Sociological Methodology, 6 (1975), pp. 59–128.

[17] C. BAJAJ, *Proving geometric algorithm non-solvability: an application of factoring polynomials*, Journal of Symbolic Computation, 2 (1986), pp. 99–102.

[18] S. BANDYOPADHYAY AND U. MAULIK, *An evolutionary technique based on K-means algorithm for optimal clustering in* $\mathbb{R}^N$, Information Sciences, 146 (2002), pp. 221–237.

[19] A. BARALDI AND E. ALPAYDIN, *Constructive feedforward art clustering networks I*, IEEE Transactions on Neural Networks, 13 (2002), pp. 645–661.

[20] ——, *Constructive feedforward art clustering networks II*, IEEE Transactions on Neural Networks, 13 (2002), pp. 662–677.

[21] A. BARALDI AND P. BLONDA, *A survey of fuzzy clustering algorithms for pattern recognition I*, IEEE Transactions on Systems, Man, and Cybernetics, Part B, 29 (1999), pp. 778–785.

[22] ——, *A survey of fuzzy clustering algorithms for pattern recognition II*, IEEE Transactions on Systems, Man, and Cybernetics, Part B, 29 (1999), pp. 786–801.

[23] V. BARNETT, *The ordering of multivariate data*, Journal of the Royal Statistical Society: Series A (General), 139 (1976), pp. 318–355.

[24] V. BARNETT AND T. LEWIS, *Outliers in statistical data*, John Wiley & Sons, 2 ed., 1984.

[25] G. BATISTA AND M. MONARD, *Experimental Comparison of k-Nearest Neighbour and Mean or Mode Imputation Methods with the Internal Strategies used by C4.5 and CN2 to Treat Missing Data*, Tech. Report 186, ICMC-USP, 2003.

[26] G. BATISTA AND M. C. MONARD, *A study of k-nearest neigbour as an imputation method*, in In Proceedins of the Second International Conference on Hybrid Intelligent Systems, Santiago, Chile, December 2002, IOS Press, pp. 251–260.

[27] G. BATISTA AND M. C. MONARD, *An analysis of four missing data treatment methods for supervised learning*, Applied Artificial Intelligence, 17 (2003), pp. 519–533.

[28] M. S. BAZARAA, H. D. SHERALI, AND C. M. SHETTY, *Nonlinear programming: Theory and algorithms*, John Wiley & Sons, Inc., 1993.

[29] A. BENSAID, L. HALL, J. BEZDEK, L. CLARKE, M. SILBIGER, J. ARRINGTON, AND R. MURTAGH, *Validity-guided (re)clustering with applications to image segmentation*, IEEE Transactions on Fuzzy Systems, 4 (1996), pp. 112–123.

[30] P. BERKHIN, *Survey of clustering data mining techniques*, tech. report, Accrue Software, San Jose, CA, 2002.

[31] M. J. BERRY AND G. S. LINOFF, *Mastering data mining: The art and science of customer relationship management*, John Wiley & Sons, Inc., 2000.

[32] M. W. BERRY, ed., *Survey of text mining: clustering, classification, and retrieval*, Springer-Verlag, Inc., 2004.

[33] D. P. BERTSEKAS, *Nonlinear Programming*, Athena Scientific, 2 ed., 1999.

[34] J. BEZDEK AND R. HATHAWAY, *VAT: a tool for visual assessment of (cluster) tendency*, in Proceedings of the International Joint Conference on Neural Networks, vol. 3, IEEE Press, May 2002, pp. 2225–2230.

[35] J. C. BEZDEK AND N. R. PAL, *Some new indexes of cluster validity*, IEEE Transactions on Systems, Man, and Cybernetics, Part B, 28 (1998), pp. 301–315.

[36] I. S. BHANDARI, E. COLET, J. PARKER, Z. PINES, R. PRATAP, AND K. RAMANUJAM, *Advanced scout: Data mining and knowledge discovery in NBA data*, Data Mining and Knowledge Discovery, 1 (1997), pp. 121–125.

[37] H. BISCHOF, A. LEONARDIS, AND A. SELB, *MDL principle for robust vector quantisation*, Pattern Analysis and Applications, 2 (1999), pp. 59–72.

[38] L. BOTTOU AND Y. BENGIO, *Convergence properties of the K-means algorithms*, in Advances in Neural Information Processing Systems, G. Tesauro, D. Touretzky, and T. Leen, eds., vol. 7, The MIT Press, 1995, pp. 585–592.

[39] C. BOUNSAYTHIP AND E. RINTA-RUNSALA, *Overview of data mining for customer behavior modeling*, Research report TTE1-2001-18, VTT Information Technology, Espoo, Finland, June 2001. Version 1.

[40] R. J. BRACHMAN AND T. ANAND, *The process of knowledge discovery in databases*, in Advances in Knowledge Discovery and Data Mining, 1996, pp. 37–57.

[41] P. BRADLEY, C. REINA, AND U. FAYYAD, *Clustering very large databases using EM mixture models*, in Proceedings of 15th International Conference on Pattern Recognition (ICPR'00), vol. 2, 2000, pp. 76–80.

[42] P. S. BRADLEY, *Data mining as an automated service*, in Advances in Knowledge Discovery and Data Mining: Proceedings of 7th Pacific-Asia Conference (PAKDD 2003), 2003, pp. 1–13.

[43] P. S. BRADLEY AND U. M. FAYYAD, *Refining initial points for K-Means clustering*, in Proceedings of the 15th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, 1998, pp. 91–99.

[44] P. S. BRADLEY, U. M. FAYYAD, AND O. L. MANGASARIAN, *Mathematical programming for data mining: formulations and challenges*, INFORMS Journal on Computing, 11 (1999), pp. 217–238.

[45] P. S. BRADLEY, U. M. FAYYAD, AND C. REINA, *Scaling clustering algorithms to large databases*, in Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, R. Agrawal and P. Stolorz, eds., AAAI Press, 1998, pp. 9–15.

[46] P. S. BRADLEY, O. L. MANGASARIAN, AND W. N. STREET, *Clustering via concave minimization*, in Advances in Neural Information Processing Systems, M. C. Mozer, M. I. Jordan, and T. Petsche, eds., vol. 9, The MIT Press, 1997, p. 368.

[47] J. BRIMBERG, *The fermat-weber location problem revisited*, Mathematical Programming, 71 (1995), pp. 71–76.

[48] J. BRIMBERG AND R. CHEN, *A note on convergence in the single facility minisum location problem*, Computers & Mathematics with Applications, 35 (1998), pp. 25–31.

[49] J. BRIMBERG, R. CHEN, AND D. CHEN, *Accelerating convergence in the fermat-weber location problem*, Operations Research Letters, 22 (1998), pp. 151–157.

[50] J. BRIMBERG AND R. F. LOVE, *Global convergence of a generalized iterative procedure for the minisum location problem with l_p distances*, Operations Research, 41 (1993), pp. 1153–1163.

[51] ——, *Local convexity results in a generalized Fermat-Weber problem*, Computers & Mathematics with Applications, 37 (1999), pp. 87–97.

[52] S. BRIN AND L. PAGE, *The anatomy of a large-scale hypertextual web search engine*, in Proceedings of the seventh International World Wide Web Conference, 1998, pp. 107–117.

[53] S. BRIN, L. PAGE, R. MOTWANI, AND T. WINOGRAD, *The pagerank citation ranking: Bringing order to the web*, Working Paper 1999-66, Stanford Digital Libraries, 1999.

[54] B. M. BROWN, *Statistical uses of the spatial median*, Journal of the Royal Statistical Society. Series B (Methodological), 45 (1983), pp. 25–30.

[55] B. M. BROWN, P. HALL, AND G. A. YOUNG, *On the effect of inliers on the spatial median*, Journal of Multivariate Analysis, 63 (1997), pp. 88–104.

[56] R. L. BURDEN AND J. D. FAIRES, *Numerical analysis*, PWS-KENT Publishing Company, Boston, 4 ed., 1989.

[57] M. C. BURL, L. ASKER, P. SMYTH, U. M. FAYYAD, P. PERONA, L. CRUMPLER, AND J. AUBELE, *Learning to recognize volcanoes on venus*, Machine Learning, 30 (1998), pp. 165–194.

[58] B. S. BUTKIEWICZ, *Robust fuzzy clustering with fuzzy data*, in Proceedings of the Third International Atlantic Web Intelligence Conference, 2005, pp. 76–82.

[59] L. CÁNOVAS, R. C. NAVATE, AND A. MARÍN, *On the convergence of the weiszfeld algorithm*, Mathematical Programming, 93 (2002), pp. 327 – 330.

[60] P. L. CARBONE, *Data mining or knowledge discovery in databases: An overview*, tech. report, 1997.

[61] S.-H. CHA, S. YOON, AND C. TAPPERT, *On binary similarity measures for handwritten character recognition*, in Proceedings of the Eighth International Conference on Document Analysis and Recognition, vol. 1, IEEE, 2005, pp. 4–8.

[62] S. CHAKRABARTI, B. DOM, R. KUMAR, P. RAGHAVAN, S. RAJAGOPALAN, A. TOMKINS, D. GIBSON, AND J. M. KLEINBERG, *Mining the web's link structure*, IEEE Computer, 32 (1999), pp. 60–67.

[63] P. K. CHAN, W. FAN, A. L. PRODROMIDIS, AND S. J. STOLFO, *Distributed data mining in credit card fraud detection*, IEEE Intelligent Systems, 14 (1999), pp. 67–74.

[64] R. CHANDRASEKARAN AND A. TAMIR, *Open questions concerning Weiszfeld's algorithm for the Fermat-Weber location problem*, Mathematical Programming, 44 (1989), pp. 293–295.

[65] J.-W. CHANG AND D.-S. JIN, *A new cell-based clustering method for large, high-dimensional data in data mining applications*, in SAC '02: Proceedings of the 2002 ACM symposium on Applied computing, New York, NY, USA, 2002, ACM Press, pp. 503–507.

[66] E. CHÁVEZ, G. NAVARRO, R. BAEZA-YATES, AND J. MARROQUIN, *Searching in metric spaces*, ACM Computing Surveys, 33 (2001), pp. 273–321.

[67] P. CHEESEMAN AND J. STUTZ, *Bayesian classification (autoclass): Theory and results*, in Advances in Knowledge Discovery and Data Mining, 1996, pp. 153–180.

[68] D. CHEN, *On two or more dimensional optimum quantizers*, in Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '77., vol. 2, Telecommunication Training Institute, Taiwan, Republic of China, May 1977, pp. 640–643.

[69] M.-S. CHEN, J. HAN, AND P. YU, *Data mining: An overview from database perspective*, IEEE Transactions on knowledge and data engineering, 8 (1996), pp. 866–883.

[70] T. CHIU, D. FANG, J. CHEN, Y. WANG, AND C. JERIS, *A robust and scalable clustering algorithm for mixed type attributes in large database environment*, in Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press, 2001, pp. 263–268.

[71] S.-C. CHU, J. F. RODDICK, AND J. S. PAN, *A comparative study and extension to K-medoids algorithms*, in Proceedings of the 5th International Conference on Optimization: Techniques and Applications (ICOTA 2001), Hong Kong, December 2001.

[72] R. CILIBRASI, P. M. B. VITÁNYI, AND R. DE WOLF, *Algorithmic clustering of music*, in WEDELMUSIC, 2004, pp. 110–117.

[73] K. J. CIOS AND L. A. KURGAN, *Trends in Data Mining and Knowledge Discovery*, Springer-Verlag, June 2005, ch. 1.

[74] F. CLARKE, *Nonsmooth Optimization and Nonsmooth Analysis*, John Wiley & Sons, New York, 1983.

[75] C. W. COAKLEY AND T. P. HETTMANSPERGER, *A bounded influence, high breakdown, efficient regression estimator*, Journal of the American Statistical Association, 88 (1993), pp. 872–880.

[76] D. COOK, L. HOLDER, S. SU, R. MAGLOTHIN, AND I. JONYER, *Structural mining of molecular biology data*, Engineering in Medicine and Biology Magazine, 20 (1999), pp. 67–74.

[77] R. COOLEY, B. MOBASHER, AND J. SRIVASTAVA, *Web mining: Information and pattern discovery on the world wide web*, in Proceedings of the Ninth International Conference on Tools with Artificial Intelligence (ICTAI '97), Newport Beach, CA, USA, November 1997, IEEE Computer Society, IEEE, pp. 558–567.

[78] L. COOPER, *Location-allocation problem*, Operations Research, 11 (1963), pp. 331–343.

[79] L. COOPER AND I. N. KATZ, *The Weber problem revisited*, An International Journal on Computers & Mathematics with Applications, 7 (1981), pp. 225–234.

[80] R. M. CORMACK, *A review of classification*, Journal of the Royal Statistical Society. Series A (General), 134 (1971), pp. 321–367.

[81] D. CORNEY, *Intelligent food analysis of small data set for food design*, PhD thesis, Department of Computer Science, University College, London, 2002.

[82] C. CROUX AND A. RUIZ-GAZEN, *High breakdown estimators for principal components: the projection-pursuit approach revisited*, Journal of Multivariate Analysis, 95 (2005), pp. 206–226.

[83] J. A. CUESTA-ALBERTOS, A. GORDALIZA, AND C. MATRÁN, *Trimmed k-means: an attempt to robustify quantizers*, The Annals of Statistics, 25 (1997), pp. 553–576.

[84] A. P. DANYLUK, F. PROVOST, AND B. CARR, *Industry: telecommunications network diagnosis*, Oxford University Press, Inc., New York, NY, USA, 2002, pp. 897–902.

[85] A. DEMPSTER, N. LAIRD, AND D. RUBIN, *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society. Series B (Methodological), 39 (1977), pp. 1–38.

[86] I. DHILLON, D. MODHA, AND W. SPANGLER, *Class visualization of high-dimensional data with applications*, Computational Statistics & Data Analysis, 41 (2002), pp. 59–90.

[87] W. R. DILLON AND M. GOLDSTEIN, *Multivariate analysis: methods and applications*, Wiley series in probability and mathematical statistics, Applied probability and statistics, Wiley, New York, 1984.

[88] S. DOLNICAR, F. LEISCH, A. WEINGESSEL, C. BUCHTA, AND E. DIMITRIADOU, *A comparison of several cluster algorithms on artificial binary data scenarios from travel market segmentation*, Working Paper 7, April 1998.

[89] D. L. DONOHO AND M. GASKO, *Breakdown properties of location estimates based on halfspace depth and projected outlyingness*, The Annals of Statistics, 20 (1992), pp. 1803–1827.

[90] Z. DREZNER, *A note on accelerating the weiszfeld procedure*, Location Science, 3 (1995), pp. 275–279.

[91] R. C. DUBES, *How many clusters are best? - an experiment*, Pattern Recognition, 20 (1987), pp. 645–663.

[92] G. R. DUCHARME AND P. MILASEVIC, *Spatial median and directional data*, Biometrika, 74 (1987), pp. 212–215.

[93] R. DUDA AND P. HART, *Pattern Classification and Scene analysis*, John Wiley & Sons, Inc., NY, 1973.

[94] R. O. DUDA, P. E. HART, AND D. G. STORK, *Pattern classification*, John Wiley & Sons, Inc., 2001.

[95] S. DUDOIT AND J. FRIDLYAND, *A prediction-based resampling method for estimating the number of clusters in a dataset*, Genome Biology, 3 (2002).

[96] M. H. DUNHAM, *Data mining - introductory and advanced topics*, Pearson Education Inc, Upper Saddle River, New Jersey, USA, 2003.

[97] M. ESTER, H.-P. KRIEGEL, J. SANDER, M. WIMMER, AND X. XU, *Incremental clustering for mining in a data warehousing environment*, in VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases, August 24-27, 1998, New York City, New York, USA, A. Gupta, O. Shmueli, and J. Widom, eds., Morgan Kaufmann, 1998, pp. 323–333.

[98] M. ESTER, H.-P. KRIEGEL, J. SANDER, AND X. XU, *A density-based algorithm for discovering clusters in large spatial databases with noise*, in Second International Conference on Knowledge Discovery and Data Mining, E. Simoudis, J. Han, and U. Fayyad, eds., Portland, Oregon, 1996, AAAI Press, pp. 226–231.

[99] V. ESTIVILL-CASTRO, *Why so many clustering algorithms: A position paper*, SIGKDD Explorations Newsletter, 4 (2002), pp. 65–75.

[100] V. ESTIVILL-CASTRO AND M. E. HOULE, *Data structures for minimization of total within-group distance for spatio-temporal clustering*, in Principles of Data Mining and Knowledge Discovery: 5th European Conference, PKDD 2001, Proceedings, 2001, pp. 91–102.

[101] ——, *Fast randomized algorithms for robust estimation of location*, Lecture Notes in Computer Science, 2007 (2001), pp. 77–88.

[102] ——, *Robust distance-based clustering with applications to spatial data mining*, Algorithmica, 30 (2001), pp. 216–242.

[103] V. ESTIVILL-CASTRO AND J. YANG, *Clustering web visitors by fast, robust and convergent algorithms*, International Journal of Foundations of Computer Science, 13 (2002), pp. 497–520.

[104] ——, *Fast and robust general purpose clustering algorithms*, Data Mining and Knowledge Discovery, 8 (2004), pp. 127–150.

[105] O. ETZIONI, *The world-wide web: Quagmire or gold mine?*, Communications of the ACM, 39 (1996), pp. 65–68.

[106] B. S. EVERITT, S. LANDAU, AND M. LEESE, *Cluster analysis*, Arnolds, a member of the Hodder Headline Group, 2001.

[107] F. M. FACCA AND P. L. LANZI, *Mining interesting knowledge from weblogs: A survey*, Data & Knowledge Engineering, 53 (2005), pp. 225–241.

[108] F. FARNSTROM, J. LEWIS, AND C. ELKAN, *Scalability for clustering algorithms revisited*, ACM SIGKDD Explorations Newsletter, 2 (2000), pp. 51–57.

[109] T. FAWCETT AND F. J. PROVOST, *Adaptive fraud detection*, Data Mining and Knowledge Discovery, 1 (1997), pp. 291–316.

[110] U. FAYYAD, D. HAUSSLER, AND P. STOLORZ, *Mining scientific data*, Communications of the ACM, 39 (1996), pp. 51–57.

[111] U. FAYYAD, G. PIATETSKY-SHAPIRO, AND P. SMYTH, *The KDD process for extracting useful knowledge from volumes of data*, Communications of the ACM, 39 (1996), pp. 27–34.

[112] U. M. FAYYAD, *SKICAT: Sky image cataloging and analysis tool*, in Proceedings of the International Joint Conference on Artificial Intelligence, 1995, pp. 2067–2068.

[113] ——, *Data mining and knowledge discovery: Making sense out of data*, IEEE Expert, 11 (1996), pp. 20–25.

[114] U. M. FAYYAD, D. HAUSSLER, AND P. E. STOLORZ, *KDD for science data analysis: Issues and examples*, in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD96, 1996, pp. 50–56.

[115] U. M. FAYYAD, G. PIATETSKY-SHAPIRO, AND P. SMYTH, *From data mining to knowledge discovery: an overview*, American Association for Artificial Intelligence, 1996, pp. 1–34.

[116] ——, *From data mining to knowledge discovery in databases*, AI Magazine, 17 (1996), pp. 37–54.

[117] ——, *Knowledge discovery and data mining: Towards a unifying framework*, in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD96, 1996, pp. 82–88.

[118] U. M. FAYYAD, G. PIATETSKY-SHAPIRO, P. SMYTH, AND R. UTHURUSAMY, eds., *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, 1996.

[119] U. M. FAYYAD, C. REINA, AND P. S. BRADLEY, *Initialization of iterative refinement clustering algorithms*, in Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD98), AAAI Press, 1998, pp. 194–198.

[120] W. D. FISHER, *On grouping for maximum homogeneity*, Journal of the American Statistical Association, 53 (1958), pp. 789–798.

[121] E. FORGY, *Cluster analysis of multivariate data: Efficiency versus interpretability of classifications*, Biometrics, 21 (1965), pp. 768–769. Abstract.

[122] E. B. FOWLKES AND C. L. MALLOWS, *A method for comparing two hierarchical clusterings*, Journal of the American Statistical Association, 78 (1983), pp. 553–569.

[123] D. FRADKIN, I. MUCHNIK, AND S. STRELTSOV, *Image compression in real-time multiprocessor systems using divisive K-means clustering*, in Proceedings of International Conference on Integration of Knowledge Intensive Multi-Agent Systems, IEEE, 2003, pp. 506–511.

[124] C. FRALEY, *Algorithms for model-based gaussian hierarchical clustering*, SIAM Journal on Scientific Computing, 20 (1998), pp. 270–281.

[125] C. FRALEY AND A. RAFTERY, *Model-based clustering, discriminant analysis, and density estimation*, Journal of the American Statistical Association, 97 (2002), pp. 611–631.

[126] C. FRALEY AND A. E. RAFTERY, *How many clusters?Which clustering method?Answers via model-based cluster analysis*, The Computer Journal, 41 (1998), pp. 578–588.

[127] W. J. FRAWLEY, G. PIATETSKY-SHAPIRO, AND C. J. MATHEUS, *Knowledge discovery in databases - an overview*, AI Magazine, 13 (1992), pp. 57–70.

[128] A. L. N. FRED AND A. K. JAIN, *Robust data clustering*, in Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03), IEEE Computer Society, IEEE, June 2003, pp. 128–136.

[129] J. FRIDLYAND AND S. DUDOIT, *Applications of resampling methods to estimate the number of clusters and to improve the accuracy of a clustering method*, Technical report 600, Department of Statistics, University of California, Berkeley, September 2001.

[130] H. P. FRIEDMAN AND J. RUBIN, *On some invariant criteria for grouping data*, Journal of the American Statistical Association, 62 (1967), pp. 1159–1178.

[131] J. FRIEDMAN, *Data mining and statistics: What's the connection?*, in Proceedings of the 29th Symposium on the Interface Between Computer Science and Statistics, Kluwer Academic Publishers, 1997.

[132] H. FRIGUI AND R. KRISHNAPURAM, *A robust competitive clustering algorithm with applications in computer vision*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 21 (1999), pp. 450–465.

[133] H. FRIGUI AND O. NASRAOUI, *Survey of Text Mining*, Springer, 2004, ch. Simultaneous Clustering and Dynamic Keyword Weighting for Text Documents, pp. 45–70.

[134] B. FRITZKE, *The LBG-U method for vector quantization - an improvement over LBG inspired from neural networks*, Neural Processing Letters, 5 (1997), pp. 35–45.

[135] K. FUKUNAGA, *Introduction to Statistical Pattern Recognition*, Academic Press, Inc, 1972.

[136] M. T. GALLEGOS AND G. RITTER, *A robust method for cluster analysis*, The Annals of Statistics, 33 (2005), pp. 347–380.

[137] V. GANTI, J. GEHRKE, AND R. RAMAKRISHNAN, *CACTUS - clustering categorical data using summaries*, in Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press, 1999, pp. 73–83.

[138] G. GARAI AND B. B. CHAUDHURI, *A novel genetic algorithm for automatic clustering*, Pattern Recognition Letters, 25 (2004), pp. 173–187.

[139] M. GAVRILOV, D. ANGUELOV, P. INDYK, AND R. MOTWANI, *Mining the stock market (extended abstract): which measure is best?*, in KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and Data mining, New York, NY, USA, 2000, ACM Press, pp. 487–496.

[140] A. GERSHO AND R. M. GRAY, *Vector Quantization and Signal Compression*, Kluwer Academic Publisher, 1992.

[141] Z. GHAHRAMANI AND M. I. JORDAN, *Learning from incomplete data*, Tech. Report AIM-1509, 1994.

[142] J. GHOSH, *Scalable clustering methods for data mining*, Lawrence Ealbaum Associates, Mahwah, New Jersey, USA, 2003, ch. 10, pp. 247–277.

[143] A. GIONIS, H. MANNILA, AND P. TSAPARAS, *Clustering aggregation*, in 21st International Conference on Data Engineering (ICDE), 2005, pp. 341–352.

[144] C. GLYMOUR, D. MADIGAN, D. PREGIBON, AND P. SMYTH, *Statistical themes and lessons for data mining*, Data Mining and Knowledge Discovery, 1 (1997), pp. 11–28.

[145] R. GNANADESIKAN, J. KETTENRING, AND S. TSAO, *Weighting and selection of variables for cluster analysis*, Journal of Classification, 12 (1995), pp. 113–136.

[146] R. GONZALEZ AND R. WOODS, *Digital image processing*, Addison-Wesley, 1993.

[147] J. GOWER, *A general coefficient of similarity and some of its properties*, Biometrics, 27 (1971), pp. 857–871.

[148] ——, *Algorithm as 78: The mediancentre*, Applied Statistics, 23 (1974), pp. 466–470.

[149] J. GRABMEIER AND A. RUDOLPH, *Techniques of cluster algorithms in data mining*, Data mining and knowledge discovery, 6 (2002), pp. 303–360.

[150] S. GUHA, R. RASTOGI, AND K. SHIM, *CURE: an efficient clustering algorithm for large databases*, in SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data, New York, NY, USA, 1998, ACM Press, pp. 73–84.

[151] ——, *ROCK: A robust clustering algorithm for categorical attributes*, Information Systems, 25 (2000), pp. 345–366.

[152] ——, *CURE: An efficient clustering algorithm for large databases*, Information Systems, 26 (2001), pp. 35–58.

[153] L. GUO, *Applying data mining techniques in property/casualty insurance*, in CAS 2003 Winter Forum, Data Management, Quality, and Technology Call Papers and Ratemaking Discussion Papers, CAS, 2003, pp. 1–26.

[154] S. K. GUPTA, K. S. RAO, AND V. BHATNAGAR, *K-means clustering algorithm for categorical attributes*, in DaWaK '99: Proceedings of the First International Conference on Data Warehousing and Knowledge Discovery, London, UK, 1999, Springer-Verlag, pp. 203–208.

[155] A. HADJIDIMOS, *Successive overrelaxation (sor) and related methods*, Journal of Computational and Applied Mathematics, 123 (2000), pp. 177–199.

[156] M. HALKIDI, Y. BATISTAKIS, AND M. VAZIRGIANNIS, *On clustering validation techniques*, Journal of Intelligent Information Systems, 17 (2001), pp. 107–145.

[157] ——, *Cluster validity methods: Part I*, SIGMOD Record, 31 (2002), pp. 40–45.

[158] ——, *Clustering validity checking methods: Part II*, SIGMOD Record, 31 (2002), pp. 19–27.

[159] M. HALKIDI AND M. VAZIRGIANNIS, *Clustering validity assessment: Finding the optimal partitioning of a data set*, in Proceedings of the 2001 IEEE International Conference on Data Mining, 2001, pp. 187–194.

[160] M. HALKIDI, M. VAZIRGIANNIS, AND Y. BATISTAKIS, *Quality scheme assessment in the clustering process*, in Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, september 2000, pp. 265–276.

[161] G. HAMERLY AND C. ELKAN, *Learning the k in k-means*, in Advances in Neural Information Processing Systems 16, S. Thrun, L. Saul, and B. Schölkopf, eds., MIT Press, Cambridge, MA, 2004.

[162] F. R. HAMPEL, *The influence curve and its role in robust estimation*, Journal of the American Statistical Association, 69 (1974), pp. 383–393.

[163] F. R. HAMPEL, E. M. RONCHETTI, P. J. ROUSSEEUW, AND W. A. STAHEL, *Robust statistics: The approach based on influence functions*, John Wiley & Sons, 1986.

[164] F. R. HAMPEL, P. J. ROUSSEEUW, AND E. RONCHETTI, *The change-of-variance curve and optimal redescending M-estimators*, Journal of the American Statistical Association, 76 (1981), pp. 643–648.

[165] J. HAN, R. ALTMAN, V. KUMAR, H. MANNILA, AND D. PREGIBON, *Emerging scientific applications in data mining*, Communications of the ACM, 45 (2002), pp. 54–58.

[166] J. HAN AND K. C.-C. CHANG, *Data mining for web intelligence*, IEEE Computer, 35 (2002), pp. 64–70.

[167] J. HAN AND M. KAMBER, *Data mining: concepts and techniques*, Morgan Kaufmann Publishers, Inc., 2001.

[168] J. HAN, M. KAMBER, AND A. K. H. TUNG, *Spatial Clustering Methods in Data Mining: A Survey*, Taylor and Francis, 1 ed., December 2001, ch. 8, pp. 188–217.

[169] J. HAN, K. KOPERSKI, AND N. STEFANOVIC, *GeoMiner: a system prototype for spatial data mining*, in SIGMOD '97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data, New York, NY, USA, 1997, ACM Press, pp. 553–556.

[170] D. HAND, H. MANNILA, AND P. SMYTH, *Principles of Data Mining*, MIT Press, 2001.

[171] D. J. HAND, *Data mining: Statistic and more?*, The American Statistician, 52 (1998), pp. 112–118.

[172] ——, *Statistics and data mining: intersecting disciplines*, ACM SIGKDD Explorations Newsletter, 1 (1999), pp. 16–19.

[173] A. HARDY, *On the number of clusters*, Computational Statistics and Data Analysis, 23 (1996), pp. 83–96.

[174] J. A. HARTIGAN, *Clustering algorithms*, John Wiley & Sons, 1975.

[175] T. HASTIE, R. TIBSHIRANI, AND J. FRIEDMAN, *The elements of statistical learning: Data mining, inference and prediction*, Springer-Verlag, 2001.

[176] R. J. HATHAWAY AND J. C. BEZDEK, *Fuzzy c-means clustering of incomplete data*, IEEE Transactions on Systems, Man, and Cybernetics, Part B, 31 (2001), pp. 735–744.

[177] ——, *Visual cluster validity for prototype generator clustering models*, Pattern Recognition Letters, 24 (2003), pp. 1563–1569.

[178] K. HÄTÖNEN, M. KLEMETTINEN, H. MANNILA, P. RONKAINEN, AND H. TOIVONEN, *Knowledge discovery from telecommunication network alarm databases*, in Proceedings of the Twelfth International Conference on Data Engineering (ICDE), IEEE Computer Society, 1996, pp. 115–122.

[179] D. M. HAWKINS AND D. J. OLIVE, *Inconsistency of resampling algorithms for high-breakdown regression estimators and a new algorithm*, Journal of the American Statistical Association, 97 (2002), pp. 136–159.

[180] S. HAWKINS, G. J. WILLIAMS, R. A. BAXTER, P. CHRISTEN, M. J. FETT, M. HEGLAND, F. HUANG, O. M. NIELSEN, T. SEMENOVA, AND A. SMITH, *Data mining of administrative claims data for pathology services*, in 34th Annual Hawaii International Conference on System Sciences (34-HICSS), vol. 6, Maui, Hawaii, 2001.

[181] S. HAYKIN, *Neural Networks: A Comprehensive Foundation*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[182] J. HE, M. LAN, C.-L. TAN, S.-Y. SUNG, AND H.-B. LOW, *Initialization of cluster refinement algorithms: A review and comparative study*, in Proceedings of International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, July 2004.

[183] Z. HE, X. XU, AND S. DENG, *Clustering mixed numeric and categorical data: A cluster ensemble approach*, tech. report, 2002.

[184] Z. HE, X. XU, S. DENG, AND Y. SONG, *dNumber: A fast clustering algorithm for very large categorical datasets*, in Proceedings of the NDBC 2002, Zheng Zhou, China.

[185] M. HEGLAND, *Data mining - challenges, models, methods and algorithms*, tech. report.

[186] E. HELMES AND J. LANDMARK, *Subtypes of schizophrenia: A cluster analytic approach*, The Canadian Journal of Psychiatry, 48 (2003), pp. 702–708.

[187] A. HINNEBURG AND D. A. KEIM, *An efficient approach to clustering in large multimedia databases with noise*, in Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 1998, AAAI Press, pp. 58–65.

[188] D. C. HOAGLIN, F. MOSTELLER, AND J. W. TUKEY, *Understanding robust and exploratory data analysis*, John Wiley & Sons, Inc., 1983.

[189] H. HRUSCHKA AND M. NATTER, *Comparing performance of feedforward neural nets and k-means for cluster-based market segmentation*, European Journal of Operational Research, 114 (1999), pp. 346–353.

[190] J. Z. HUANG, M. K. NG, H. RONG, AND Z. LI, *Automated variable weighting in k-means type clustering*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 27 (2005), pp. 657–668.

[191] Z. HUANG, *A fast clustering algorithm to cluster very large categorical data sets in data mining*, in DMKD'97 Pre-Conference Data Mining Workshop: Research Issues on Data Mining and Knowledge Discovery, 1997.

[192] ——, *Extensions to the K-means algorithm for clustering large data sets with categorical values*, Data Mining and Knowledge Discovery, 2 (1998), pp. 283–304.

[193] Z. HUANG, A. ELIËNS, A. VAN BALLEGOOIJ, AND P. D. BRA, *A taxonomy of web agents*, in 11th International Workshop on Database and Expert Systems Applications (DEXA), 2000, pp. 765–769.

[194] J. M. HUBAND, J. C. BEZDEK, AND R. J. HATHAWAY, *bigVAT: Visual assessment of cluster tendency for large data sets*, Pattern Recognition, 38 (2005), pp. 1875–1886.

[195] P. HUBER, *Robust statistics*, John Wiley & Sons, 1981.

[196] P. J. HUBER, *Robust estimation for a location parameter*, The Annals of Mathematical Statistics, 35 (1964), pp. 73–101.

[197] P. J. HUBER, *The 1972 Wald lecture. Robust statistics: A review*, The Annals of Mathematical Statistics, 43 (1972), pp. 1041–1067.

[198] ——, *Finite sample breakdown of M- and P-estimators*, The Annals of Statistics, 12 (1984), pp. 119–126.

[199] ——, *John W. Tukey's contributions to robust statistics*, The Annals of Statistics, 30 (2002), pp. 1640–1648. In memory of John W. Tukey.

[200] A. HYVÄRINEN, J. KARHUNEN, AND E. OJA, *Independent component analysis*, Wiley Interscience, 2001.

[201] T. IMIELINSKI AND H. MANNILA, *A database perspective on knowledge discovery*, Communications of the ACM, 39 (1996), pp. 58–64.

[202] T. I. S. B. S. G. (ISBSG), *Estimating, Benchmarking and Research Suite Release 9*. CD ROM, 2004.

[203] A. JAIN, M. MURTY, AND P. FLYNN, *Data clustering: a review*, ACM Computing Surveys, 31 (1999), pp. 264–323.

[204] A. K. JAIN AND R. C. DUBES, *Algorithms for clustering data*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[205] A. K. JAIN, R. P. W. DUIN, AND J. MAO, *Statistical pattern recognition: A review*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (2000), pp. 4–37.

[206] D. JIANG, C. TANG, AND A. ZHANG, *Cluster analysis for gene expression data: A survey*, IEEE Transactions on Knowledge and Data Engineering, 16 (2004), pp. 1370–1386.

[207] R. JÖRNSTEN, Y. VARDI, AND C.-H. ZHANG, *A Robust Clustering Method and Visualization Tool Based on Data Depth*, Birkhäuser Verlag, Switzerland, 2002, pp. 67–76.

[208] T. KÄRKKÄINEN AND S. ÄYRÄMÖ, *Robust clustering methods for incomplete and erroneous data*, in Proceedings of the Fifth Conference on Data Mining, 2004, pp. 101–112.

[209] ——, *On computation of spatial median for robust data mining*, in Proceedings of Sixth Conference on Evolutionary and Deterministic Methods for Design, Optimisation and Control with Applications to Industrial and Societal Problems (EUROGEN 2005), R. Schilling, W. Haase, J. Periaux, and H. Baier, eds., 2005.

[210] T. KÄRKKÄINEN, S. ÄYRÄMÖ, AND T. KILPELÄINEN, *Data Mining: Osaraportit II*, 2003.

[211] T. KÄRKKÄINEN, S. ÄYRÄMÖ, T. KILPELÄINEN, AND K. LAHTI, *Data Mining: Osaraportit I*, 2003.

[212] T. KÄRKKÄINEN, S. ÄYRÄMÖ, M. NURMINEN, AND R. SUVINEN, *Knowledge Mining -projekti: Raportit*, 2004.

[213] T. KÄRKKÄINEN AND E. HEIKKOLA, *Robust formulations for training multilayer perceptrons*, Neural Computation, 16 (2004), pp. 837–862.

[214] T. KÄRKKÄINEN AND K. MAJAVA, *Nonmonotone and monotone active-set methods for image restoration, part 1: convergence analysis*, Journal of Optimization Theory and Applications, 106 (2000), pp. 61–80.

[215] ——, *Nonomotone and monoton active-set methods for image restoration, part 2: numerical results*, Journal of Optimization Theory and Applications, 106 (2000), pp. 81–105.

[216] T. KÄRKKÄINEN, K. MAJAVA, AND M. M. MÄKELÄ, *Comparison of formulations and solution methods for image restoration problems*, Inverse Problems, 17 (2001), pp. 1977–1995.

[217] G. KARYPIS, E.-H. HAN, AND V. KUMAR, *CHAMELEON: Hierarchical clustering using dynamic modeling*, Computer, 32 (1999), pp. 68–75.

[218] I. KATSAVOUNIDIS, C.-C. JAY KUO, AND Z. ZHANG, *A new initialization technique for generalized Lloyd iteration*, Signal Processing Letters, 1 (1994), pp. 144–146.

[219] I. N. KATZ, *Local convergence in fermat's problem*, Mathematical Programming, 6 (1974), pp. 89–104.

[220] L. KAUFMAN AND P. J. ROUSSEEUW, *Finding groups in data: An introduction to cluster analysis*, John Wiley & Sons, 1990.

[221] E. J. KEOGH AND S. KASETTY, *On the need for time series data mining benchmarks: A survey and empirical demonstration*, Data Mining and Knowledge Discovery, 7 (2003), pp. 349–371.

[222] S. KHAN AND A. AHMAD, *Cluster center initialization algorithm for k-means clustering*, Pattern Recognition Letters, 25 (2004), pp. 1293–1302.

[223] T. KILPELÄINEN AND P. TYRVÄINEN, *The degree of digitalization of the information over-flow: A case study*, in Proceedings of the 5th International Conference on Enterprise Information Systems, vol. 3, 2004, pp. 367–374.

[224] D.-W. KIM, K. LEE, D. LEE, AND K. H. LEE, *A kernel-based subtractive clustering method*, Pattern Recognition Letters, 26 (2005), pp. 879–891.

[225] M. KIM AND R. S. RAMAKRISHNA, *New indices for cluster validity assessment*, Pattern Recognition Letters, 26 (2005), pp. 2353–2363.

[226] W. KIM, B.-J. CHOI, E.-K. HONG, S.-K. KIM, AND D. LEE, *A taxonomy of dirty data*, Data Mining and Knowledge Discovery, 7 (2003), pp. 81–99.

[227] Y. KIM AND W. N. STREET, *An intelligent system for customer targeting: a data mining approach*, Decision Support Systems, 37 (2004), pp. 215–228.

[228] A. KITAMOTO, *Spatio-temporal data mining for typhoon image collection*, Journal of Intelligent Information Systems, 19 (2002), pp. 25–41.

[229] M. KLEMETTINEN, *A Knowledge Discovery Methodology for Telecommunication Network Alarm Databases*, PhD thesis, University of Helsinki, Finland, 1999.

[230] M. KLEMETTINEN, H. MANNILA, AND H. TOIVONEN, *Interactive exploration of interesting findings in the telecommunication network alarm sequence analyzer TASA*, Information & Software Technology, 41 (1999), pp. 557–567.

[231] R. KOHAVI, *Mining e-commerce data: The good, the bad, and the ugly*, in Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD).

[232] R. KOHAVI, B. M. MASAND, M. SPILIOPOULOU, AND J. SRIVASTAVA, *Web mining*, Data Mining and Knowledge Discovery, 6 (2002), pp. 5–8.

[233] R. KOHAVI AND F. J. PROVOST, *Applications of data mining to electronic commerce*, Data Mining and Knowledge Discovery, 5 (2001), pp. 5–10.

[234] R. KOHAVI, N. J. ROTHLEDER, AND E. SIMOUDIS, *Emerging trends in business analytics*, Communications of the ACM, 45 (2002), pp. 45–48.

[235] R. KOSALA AND H. BLOCKEEL, *Web mining research: a survey*, SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, 2 (2000), pp. 1–15.

[236] R. KOTHARI AND D. PITTS, *On finding the number of clusters*, Pattern Recognition Letters, 20 (1999), pp. 405–416.

[237] B. KÖVESI, J.-M. BOUCHER, AND S. SAOUDI, *Stochastic k-means algorithm for vector quantization*, Pattern Recognition Letters, 22 (2001), pp. 603–610.

[238] H.-P. KRIEGEL, P. KRÖGER, AND I. GOTLIBOVICH, *Incremental OPTICS: Efficient computation of updates in a hierarchical cluster ordering*, in Proceedings of the 5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'03), 2003, pp. 101–112.

[239] K. KRISHNA AND M. NARASIMHA MURTY, *Genetic K-means algorithm*, IEEE Transactions on Systems, Man and Cybernetics, Part B, 29 (1999), pp. 433–439.

[240] M. KŘÍZEK, P. NEITTAANMÄKI, R. GLOWINSKI, AND S. KOROTOV, eds., *Conjugate Gradient Algorithms and Finite Element Methods*, Scientific Computation, Berlin Heidelberg, 2004, Springer-Verlag.

[241] H. W. KUHN, *A note on Fermat's problem*, Mathematical programming, 4 (1973), pp. 98–107.

[242] J. LAGARIAS, J. A. REEDS, M. H. WRIGHT, AND P. E. WRIGHT, *Convergence properties of the Nelder-Mead simplex method in low dimensions*, SIAM Journal of Optimization, 9 (1998), pp. 112–147.

[243] T. LANGE, V. ROTH, M. L. BRAUN, AND J. M. BUHMANN, *Stability-based validation of clustering solutions*, Neural Computation, 16 (2004), pp. 1299–1323.

[244] M. LAW, A. TOPCHY, AND A. JAIN, *Multiobjective data clustering*, in Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, 2004, pp. 424–430.

[245] M. LAWRENCE AND J. OSTRESH, *On the convergence of a class of iterative methods for solving the weber location problem*, Operations research, 26 (1978), pp. 597–609.

[246] R. D. LAWRENCE, G. S. ALMASI, V. KOTLYAR, M. S. VIVEROS, AND S. DURI, *Personalization of supermarket product recommendations*, Data Mining and Knowledge Discovery, 5 (2001), pp. 11–32.

[247] W. LEE, S. J. STOLFO, E. ESKIN, M. MILLER, S. HERSHKOP, J. ZHANG, P. K. CHAN, AND W. FAN, *Real time data mining-based intrusion detection*, in DARPA Information Survivability Conference and Exposition (DISCEX II'01), vol. 1, 2001.

[248] S. LÉTOURNEAU, F. FAMILI, AND S. MATWIN, *Data mining to predict aircraft component replacement*, IEEE Intelligent Systems, 14 (1999), pp. 59–66.

[249] Y. LEVIN AND A. BEN-ISRAEL, *The Newton bracketing method for convex minimization*, Computational Optimization and Applications, 21 (2002), pp. 213–229.

[250] E. LEVINE AND E. DOMANY, *Resampling method for unsupervised estimation of cluster validity*, Neural Computation, 13 (2001), pp. 2573–2593.

[251] R. M. LEWIS, V. TORCZON, AND M. W. TROSSET, *Direct search methods: Then and now*, Tech. Report 26, Institute for Computer Applications in Science and Engineering (NASA Langley Research Center), Hampton, Virginia, May 2000.

[252] Q. LI, C. FRALEY, R. E. BUMGARNER, K. Y. YEUNG, AND A. E. RAFTERY, *Donuts, scratches and blanks: robust model-based segmentation of microarray images*, Bioinformatics, 21 (2005), pp. 2875–2882.

[253] Y. LI, *A Newton acceleration of the Weiszfeld algorithm for minimizing the sum of Euclidean distances*, Computational Optimization and Applications, 10 (1998), pp. 219–242.

[254] A. LIKAS, N. VLASSIS, AND J. J. VERBEEK, *The global k-means clustering algorithm*, Pattern Recognition, 36 (2003), pp. 451–461.

[255] Y. LINDE, A. BUZO, AND R. GRAY, *An algorithm for vector quantizer design*, IEEE Transactions on Communications, 28 (1980), pp. 84–95.

[256] R. L. LING, *A computer generated aid for cluster analysis*, Communications of the ACM, 16 (1973), pp. 355–361.

[257] B. B. LITTLE, W. L. JOHNSTON, A. C. LOVELL, R. M. REJESUS, AND S. A. STEED, *Collusion in the U.S. crop insurance program: applied data mining*, in KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, 2002, ACM Press, pp. 594–598.

[258] R. J. LITTLE AND D. B. RUBIN, *Statistical analysis with missing data*, John Wiley & Sons, 1987.

[259] J. LIU, J. P. LEE, L. LI, Z.-Q. LUO, AND K. M. WONG, *Online clustering algorithms for radar emitter classification*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 27 (2005), pp. 1185–1196.

[260] S. P. LLOYD, *Least squares quantization in PCM*, IEEE Transactions on Information Theory, 28 (1982), pp. 129–136.

[261] H. P. LOPUHAÄ AND P. J. ROUSSEEUW, *Breakdown points of affine equivariant estimators of multivariate location and covariance matrices*, The Annals of Statistics, 19 (1991), pp. 229–248.

[262] R. LOVE, J. MORRIS, AND G. WESOLOWSKY, *Facilities Location. Models and Methods*, North Holland Publishing Company, 1988.

[263] Y. LU, S. LU, F. FOTOUHI, Y. DENG, AND S. J. BROWN, *FGKA: a fast genetic K-means clustering algorithm*, in SAC '04: Proceedings of the 2004 ACM symposium on Applied computing, ACM Press, 2004, pp. 622–623.

[264] M.-C. LUDL AND G. WIDMER, *Density-based centroid approximation for initializing iterative clustering algorithms*, tech. report.

[265] J. MACQUEEN, *Some methods for classification and analysis of multivariate observations*, in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.

[266] R. MAITRA, *Clustering massive datasets with applications in software metrics and tomography*, Technometrics, 43 (2001), pp. 336–346.

[267] M. M. MÄKELÄ AND P. NEITTAANMÄKI, *Nonsmooth Optimization; Analysis and Algorithms with Applications to Optimal Control*, World Scientific, Singapore, 1992.

[268] G. MANCO, E. MASCIARI, M. RUFFOLO, AND A. TAGARELLI, *Towards an adaptive mail classifier*, tech. report, Italian Association for Artificial Intelligence, 2002.

[269] O. L. MANGASARIAN AND D. R. MUSICANT, *Successive overrelaxation for support vector machines*, IEEE Transactions On Neural Networks, 10 (1999), pp. 1032–1037.

[270] H. MANNILA, *Methods and problems in data mining*, in Proceedings of International Conference on Database Theory (ICDT), F. Afrati and P. Kolaitis, eds., Springer-Verlag, January 1997, pp. 41–55.

[271] H. MANNILA, H. TOIVONEN, AND A. I. VERKAMO, *Discovery of frequent episodes in event sequences*, Data Mining and Knowledge Discovery, 1 (1997), pp. 259–289.

[272] J. MARDEN, *Some robust estimates of principle components*, Probability and Statistics Letters, 43 (1999), pp. 349–359.

[273] M. MARINA AND H. DAVID, *An experimental comparison of several clustering and initialization methods*, in Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98), San Francisco, CA, 1998, Morgan Kaufmann Publishers, pp. 386–395.

[274] R. A. MARONNA, *Robust M-estimators of multivariate location and scatter*, The Annals of Statistics, 4 (1976), pp. 51–67.

[275] F. H. C. MARRIOTT, *Practical problems in a method of cluster analysis*, Biometrics, 27 (1971), pp. 501–514.

[276] J.-C. MASSÉ AND J.-F. PLANTE, *A Monte Carlo study of the accuracy and robustness of ten bivariate location estimators*, Computational Statistics & Data Analysis, 42 (2003), pp. 1–26.

[277] U. MAULIK AND S. BANDYOPADHYAY, *Performance evaluation of some clustering algorithms and validity indices*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 24 (2002), pp. 1650–1654.

[278] P. MEER, *Robust Techniques for Computer Vision*, Prentice Hall PTR, 2004, ch. 4.

[279] W. MIEHLE, *Link-length minimization in networks*, Operations research, 6 (1958), pp. 232–243.

[280] K. MIETTINEN, *Nonlinear Multiobjective Optimization*, Kluwer Academic Publishers, Boston, 1999.

[281] P. MILASEVIC AND G. R. DUCHARME, *Uniqueness of the spatial median*, The Annals of Statistics, 15 (1987), pp. 1332–1333.

[282] R. G. MILLER, *The jackknife—a review*, Biometrika, 61 (1974), pp. 1–15.

[283] G. MILLIGAN AND M. COOPER, *An examination of procedures for determining the number of clusters in a data set*, Psychometrika, 50 (1985), pp. 159–179.

[284] G. W. MILLIGAN AND M. C. COOPER, *A study of standardization of variables in cluster analysis*, Journal of Classification, 5 (1988), pp. 181–204.

[285] S. MITRA, *An evolutionary rough partitive clustering*, Pattern Recognition Letters, 25 (2004), pp. 1439–1449.

[286] B. MOBASHER, R. COOLEY, AND J. SRIVASTAVA, *Creating adaptive Web sites through usage-based clustering of URLs*, in Proceedings 1999 Workshop on Knowledge and Data Engineering Exchange (KDEX '99), Los Alamitos, CA, USA, 2000, IEEE, pp. 19–25.

[287] F. MÖRCHEN AND A. ULTSCH, *Mining hierarchical temporal patterns in multivariate time series*, in KI 2004: Advances in Artificial Intelligence, Proceedings of 27th Annual German Conference in AI, Springer Heidelberg, 2004, pp. 127–140.

[288] J. G. MORRIS, *Convergence of the weiszfeld algorithm for weber problems using a generalized "distance" function*, Operations Research, 29 (1981), pp. 37–48.

[289] J. G. MORRIS AND W. A. VERDINI, *Minisum l_p distance location problems solved via a perturbed problem and Weiszfeld's algorithm*, Operations Research, 27 (1979), pp. 1180–1188.

[290] J. MÖTTÖNEN AND H. OJA, *Multivariate spatial sign and rank methods*, Journal of Nonparametric Statistics, 5 (1995), pp. 201–213.

[291] G. B. MUFTI, P. BERTRAND, AND L. E. MOUBARKI, *Determining the number of groups from measures of cluster stability*, in Proceedings of International Symposium on Applied Stochastic Models and Data Analysis (ASMDA 2005), May 2005, pp. 404–413.

[292] I. MYRTVEIT, E. STENSRUD, AND U. H. OLSSON, *Analyzing data sets with missing data: An empirical evaluation of imputation methods and likelihood-based methods*, IEEE Transactions on Software Engineering, 27 (2001), pp. 999–1013.

[293] E. NAKAMURA AND N. D. KEHTARNAVAZ, *Determining number of clusters and prototype locations via multi-scale clustering*, Pattern Recognition Letters, 19 (1998), pp. 1265–1283.

[294] O. NASRAOUI, *Encyclopedia of Data Mining and Data Warehousing*, Idea Group, 2005, ch. World Wide Web Personalization.

[295] O. NASRAOUI AND R. KRISHNAPURAM, *A new evolutionary approach to web usage and context sensitive associations mining*, International Journal of Computational Intelligence and Applications, Special Issue on Internet Intelligent Systems, 2 (2002), pp. 339–348.

[296] O. NASRAOUI, R. KRISHNAPURAM, A. JOSHI, AND T. KAMDAR, *E-Commerce and Intelligent Methods*, Studies in Fuzziness and Soft Computing, Springer-Verlag, 2002, ch. Automatic Web User Profiling and Personalization using Robust Fuzzy Relational Clustering.

[297] G. NAVARRO, *Searching in metric spaces by spatial approximation*, The VLDB Journal, 11 (2002), pp. 28–46.

[298] L. NAZARETH AND P. TSENG, *Gilding the lily: A variant of the Nelder-Mead algorithm based on golden-section search*, Computational Optimization and Applications, 22 (2002), pp. 133–144.

[299] Z. NAZERI, E. BLOEDORN, AND P. OSTWALD, *Experiences in mining aviation safety data*, in SIGMOD '01: Proceedings of the 2001 ACM SIGMOD international conference on Management of data, New York, NY, USA, 2001, ACM Press, pp. 562–566.

[300] J. NELDER AND R. MEAD, *A simplex method for function minimization*, Computer Journal, 7 (1965), pp. 308–313.

[301] M. K. NG, Z. HUANG, AND M. HEGLAND, *Data-mining massive time series astronomical data sets - a case study*, in Research and Development in Knowledge Discovery and Data Mining, Proceedings of the Second Pacific-Asia Conference, PAKDD-98, 1998, pp. 401–402.

[302] R. T. NG AND J. HAN, *Efficient and effective clustering methods for spatial data mining*, in Proceedings of 20th International Conference on Very Large Data Bases, J. Bocca, M. Jarke, and C. Zaniolo, eds., Los Altos, CA 94022, USA, september 1994, Morgan Kaufmann Publishers, pp. 144–155.

[303] R. T. NG AND J. HAN, *CLARANS: A method for clustering objects for spatial data mining*, IEEE Transactions on Knowledge and Data Engineering, 14 (2002), pp. 1003–1016.

[304] S. NITTEL, K. W. NG, AND R. R. MUNTZ, *CONQUEST: CONcurrent QUEries over Space and Time*, in Integrated Spatial Databases, 1999, pp. 286–307.

[305] J. NOCEDAL AND S. WRIGHT, *Numerical Optimization*, Springer-Verlag, 1999.

[306] K.-L. ONG, Z. ZHANG, W.-K. NG, AND E.-P. LIM, *Agents and stream data mining: A new perspective*, IEEE Intelligent Systems, 20 (2005), pp. 60–67.

[307] C. ORDONEZ, *Clustering binary data streams with K-means*, in DMKD '03: Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery, New York, NY, USA, 2003, ACM Press, pp. 12–19.

[308] C. ORDONEZ AND E. OMIECINSKI, *FREM: fast and robust EM clustering for large data sets*, in CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management, New York, NY, USA, 2002, ACM Press, pp. 590–599.

[309] C. ORDONEZ AND E. OMIECINSKI, *Efficient disk-based K-means clustering for relational databases*, IEEE Transactions on Knowledge and Data Engineering, 16 (2004), pp. 909–921.

[310] G. PANDURANGAN, P. RAGHAVAN, AND E. UPFAL, *Using pagerank to characterize web structure*, in COCOON '02: Proceedings of the 8th Annual International Conference on Computing and Combinatorics, London, UK, 2002, Springer-Verlag, pp. 330–339.

[311] P. M. PARDALOS AND H. E. ROMEIJN, eds., *Handbook of Global Optimization*, vol. 2, Kluwer Academic Publishers, 2002.

[312] M. PAVAN AND M. PELILLO, *A new graph-theoretic approach to clustering and segmentation*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2003, pp. 145–152.

[313] D. PELLEG AND A. MOORE, *Accelerating exact k-means algorithms with geometric reasoning*, in Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, S. Chaudhuri and D. Madigan, eds., New York, NY, August 1999, AAAI Press, pp. 277–281. An extended version is available as Technical Report CMU-CS-00-105.

[314] D. PELLEG AND A. W. MOORE, *X-means: Extending K-means with efficient estimation of the number of clusters*, in ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning, San Francisco, CA, USA, 2000, Morgan Kaufmann Publishers Inc., pp. 727–734.

[315] J. M. PENA, J. A. LOZANO, AND P. LARRANAGA, *An empirical comparison of four initialization methods for the K-means algorithm*, Pattern Recognition Letters, 20 (1999), pp. 1027–1040.

[316] G. PIATETSKY-SHAPIRO, *Knowledge discovery in real databases: A report on the IJCAI-89 workshop*, AI Magazine, 11 (1991), pp. 68–70.

[317] ——, *The data-mining industry coming of age*, IEEE Intelligent Systems, 14 (1999), pp. 32–34.

[318] G. PIATETSKY-SHAPIRO, C. MATHEUS, P. SMYTH, AND R. UTHURUSAMY, *KDD93: Progress and challenges in knowledge discovery in databases*, AI Magazine, (1994), pp. 77–82.

[319] D. PIERRAKOS, G. PALIOURAS, C. PAPATHEODOROU, AND C. D. SPYROPOULOS, *Web usage mining as a tool for personalization: A survey*, User Modeling and User-Adapted Interaction, 13 (2003), pp. 311–372.

[320] T. PÄIVÄRINTA, *A Genre-Based Approach to Developing Electronic Document Management in the Organization*, PhD thesis, University of Jyväskylä, 2001.

[321] J. PODANI, *Extending Gower's general coefficient of similarity to ordinal characters*, Taxon, 48 (1999), pp. 331–340.

[322] L. PORTNOY, E. ESKIN, AND S. J. STOLFO, *Intrusion detection with unlabeled data using clustering*, in Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001), Philadelphia, PA, November 2001.

[323] W. PRESS, B. FLANNERY, S. TENKOLSKY, , AND W. WETTERING, *Numerical Recipes in C*, Campridge University Press, Cambridge and New York, 1988.

[324] D. PYLE, *Data preparation for data mining*, Morgan Kaufmann Publishers, Inc., 2001.

[325] Y. QIAN AND C. SUEN, *Clustering combination method*, in Proceedings of 15th International Conference on Pattern Recognition (ICPR'00), vol. 2, IEEE Computer Society, September 2000, pp. 732–735.

[326] H. RALAMBONDRAINY, *A conceptual version of the K-means algorithm*, Pattern Recognition Letters, 16 (1995), pp. 1147–1157.

[327] L. RAMASWAMY, B. GEDIK, AND L. LIU, *A distributed approach to node clustering in decentralized peer-to-peer networks*, IEEE Transactions on Parallel and Distributed Systems, 16 (2005), pp. 814–829.

[328] D. RAUTENBACH, M. STRUZYNA, C. SZEGEDY, AND J. VYGEN, *Weiszfeld's algorithm revisited once again*, tech. report, Research Institute for Discrete Mathematics, University of Bonn, 2004.

[329] S. RAY AND R. H. TURI, *Determination of number of clusters in K-means clustering and application in colour image segmentation*, in Proceedings of the 4th International Conference on Advances in Pattern Recognition and Digital Techniques (ICAPRDT'99), New Delhi, India, December 1999, Narosa Publishing House, pp. 137–143.

[330] H. W. RESSOM, D. WANG, AND P. NATARAJAN, *Adaptive double self-organizing maps for clustering gene expression profiles*, Neural Networks, 16 (2003), pp. 633–640.

[331] S. J. ROBERTS, R. EVERSON, AND I. REZEK, *Minimum entropy data partitioning*, in Proceedings of International Conference on Artificial Neural Networks, vol. 2, 1999, pp. 844–849.

[332] R. T. ROCKAFELLAR, *Convex analysis*, Princeton University Press, New Jersey, 1970.

[333] D. ROCKE AND D. L. WOODRUFF, *Robust estimation of multivariate location and shape*, Journal of Statistical Planning and Inference, 57 (1997), pp. 245–255.

[334] D. M. ROCKE AND J. J. DAI, *Sampling and subsampling for cluster analysis in data mining: With applications to sky survey data*, Data Mining and Knowledge Discovery, 7 (2003), pp. 215–232.

[335] J. B. ROSEN AND G. L. XUE, *On the convergence of a hyperboloid approximation procedure for the perturbed Euclidean multifacility location problem*, Operations Research, 41 (1993), pp. 1164–1171.

[336] V. ROTH, M. BRAUN, J. BUHMANN, AND T. LANGE, *A resampling approach to cluster validation*, in COMPSTAT 2002 - Proceedings in Computational Statistics, W. Härdle and B. Rönz, eds., Heidelberg, 2002, Physica-Verlag, pp. 123–128.

[337] P. J. ROUSSEEUW, *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis*, Journal of Computational and Applied Mathematics, 20 (1987), pp. 53–65.

[338] P. J. ROUSSEEUW AND A. M. LEROY, *Robust regression and outlier detection*, John Wiley & Sons, Inc., 1987.

[339] S. SAALASTI, *Neural Networks for heart rate time series analysis*, PhD thesis, University of Jyväskylä, 2003.

[340] S. SALVADOR AND P. CHAN, *Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms*, in Proceedings Sixteenth IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2004, Los Alamitos, CA, USA, 2004, IEEE Computer Society, pp. 576–584.

[341] O. SAN, V. HUYNH, AND Y. NAKAMORI, *An alternative extension of the k-means algorithm for clustering categorical data*, International Journal of Applied Mathematics and Computer Science, 14 (2004), pp. 241–247.

[342] J. SANDER, M. ESTER, H.-P. KRIEGEL, AND X. XU, *Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications*, Data Mining and Knowledge Discovery, 2 (1998), pp. 169–194.

[343] R. SASISEKHARAN, V. SESHADRI, AND S. M. WEISS, *Data mining and forecasting in large-scale telecommunication networks*, IEEE Expert: Intelligent Systems and Their Applications, 11 (1996), pp. 37–43.

[344] S. SELIM AND M. ISMAIL, *K-means-type algorithms: A generalized convergence theorem and characterization of local optimality*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 6 (1984), pp. 81–87.

[345] P. P. SENELLART AND V. D. BLONDEL, *Automatic discovery of similar words*, Springer-Verlag, Inc., 2004, ch. 2.

[346] A. Y. SEYDIM, *Intelligent agents: a data mining perspective*, tech. report, Department of Computer Science and Engineering, Southern Methodist University, Dallas, April 1999.

[347] A. SHADEMAN AND M. ZIA, *Adaptive vector quantization of MR images using on-line K-means algorithm*, in Proceedings of SPIE, 46th Annual Meeting, Application of Digital Image Processing XXIV Conference, vol. 4472, 2001, pp. 463–470.

[348] G. SHEIKHOLESLAMI, S. CHATTERJEE, AND A. ZHANG, *WaveCluster: A multi-resolution clustering approach for very large spatial databases*, in Proceedings of 24rd International Conference on Very Large Data Bases, 1998, pp. 428–439.

[349] F. SILVESTRI, R. BARAGLIA, P. PALMERINI, AND M. SERRANÓ, *On-line generation of suggestions for web users*, in Proceedings of the International Conference on Information Technology: Coding and Computing, 2004, pp. 392–397.

[350] E. SIMOUDIS, *Reality check for data mining*, IEEE Expert, 11 (1996), pp. 26–33.

[351] P. SIMPSON, *Fuzzy min-max neural networks – Part 2: Clustering*, IEEE Transactions on Fuzzy Systems, 1 (1993), pp. 32–44.

[352] C. SMALL, *A survey on multidimensional medians*, International Statistical Review, 58 (1990), pp. 263–277.

[353] P. SMYTH, *Clustering using Monte Carlo cross-validation*, in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), AAAI Press, 1996, pp. 126–133.

[354] P. SMYTH, *Data mining at the interface of computer science and statistics*, in Data mining for scientific and engineering application, Kluwer Academic Publishers, 2001, pp. 35–62.

[355] M. SPILIOPOULOU AND C. POHLE, *Data mining for measuring and improving the success of web sites*, Data Mining and Knowledge Discovery (special issue on "E-Commerce"), 5 (2001), pp. 85–114.

[356] C. SPINUZZI, *Grappling with distributed usability: A cultural-historical examination of documentation genres over four decades*, Technical Writing and Communication, 31 (2001), pp. 41–59.

[357] P. SPRENT, *Data driven statistical methods*, Chapman & Hall, 1998.

[358] J. SRIVASTAVA, R. COOLEY, M. DESHPANDE, AND P.-N. TAN, *Web usage mining: Discovery and applications of usage patterns from web data*, SIGKDD Explorations, 1 (2000), pp. 12–23.

[359] D. STENMARK, *Information vs. knowledge: The role of intranets in knowledge management*, in Proceedings of the 35th Hawaii International Conference on System Sciences, IEEE, January 2002.

[360] S. M. STIGLER, *Do robust estimators work with real data?*, The Annals of Statistics, 5 (1977), pp. 1055–1098. With discussion and a reply by the author.

[361] S. STILL AND W. BIALEK, *How many clusters? An information-theoretic perspective*, Neural Computation, 16 (2004), pp. 2483–2506.

[362] P. E. STOLORZ AND C. DEAN, *Quakefinder: A scalable data mining system for detecting earthquakes from space*, in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96.

[363] A. STREHL AND J. GHOSH, *Relationship-based clustering and visualization for high-dimensional data mining*, INFORMS Journal on Computing, (2002), pp. 208–230.

[364] A. STRUYF, M. HUBERT, AND P. J. ROUSSEEUW, *Integrating robust clustering techniques in S-PLUS*, Computational Statistics & Data Analysis, 26 (1997), pp. 17–37.

[365] T. SU AND J. G. DY, *A deterministic method for initializing K-means clustering*, in 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'04), 2004, pp. 784–786.

[366] C. SUGAR AND G. JAMES, *Finding the number of clusters in a data set : An information theoretic approach*, Journal of the American Statistical Association, 98 (2003), pp. 750–763.

[367] S. SÜSSTRUNK, R. BUCKLEY, AND S. SWEN, *Standard RGB Color Spaces*, in Proceedings of IS&T/SID 7th Color Imaging Conference, vol. 7, 1999, pp. 127–134.

[368] J. SWALES, *Genre Analysis: English in Academic and Research Settings*, Cambridge UP, Cambridge, 1990.

[369] P.-N. TAN, M. STEINBACH, AND V. KUMAR, *Introduction to data mining*, Addison-Wesley, 2005.

[370] R. TIBSHIRANI, G. WALTHER, D. BOTSTEIN, AND P. BROWN, *Cluster validation by prediction strength*, tech. report, Department of Biostatistics, Stanford University, September 2001.

[371] R. TIBSHIRANI, G. WALTHER, AND T. HASTIE, *Estimating the number of clusters in a data set via the gap statistic*, Journal of the Royal Statistical Society: Series B (Statistical Methodology), 63 (2001), pp. 411–423.

[372] J. TOU AND R. GONZALEZ, *Pattern Recognition Principles*, Addison-Wesley Publishing Co., Reading, Massachusetts, 1974.

[373] G. C. TSENG AND W. H. WONG, *Tight clustering: A resampling-based approach for identifying stable and tight patterns in data*, Biometrics, 61 (2005), pp. 10–16.

[374] J. TUKEY, *Exploratory Data Analysis*, Addison-Wesley, 1977.

[375] J. W. TUKEY, *We need both exploratory and confirmatory*, The American Statistician, 34 (1980), pp. 23–25.

[376] A. K. H. TUNG, R. T. NG, L. V. S. LAKSHMANAN, AND J. HAN, *Constraint-based clustering in large databases*, in Proceedings of 2001 International Conference on Database Theory, 2001, pp. 405–419.

[377] I. TUOMI, *Data is more than knowledge: Implications of the reversed knowledge hierarchy for knowledge management and organizational memory*, Journal of Management Information Systems, 16 (1999), pp. 107–121.

[378] P. TYRVÄINEN, *Estimating applicability of new mobile content formats to organizational use*, in Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS-36 2003), IEEE Computer Society, 2003, p. 295.

[379] P. TYRVÄINEN, T. KILPELÄINEN, AND M. JÄRVENPÄÄ, *Patterns and measures of digitalisation in business unit communication*, International Journal of Business Information Systems, 1 (2005), pp. 199–219.

[380] H. ÜSTER AND R. F. LOVE, *The convergence of the weiszfeld algorithm*, Computers & Mathematics with Applications, 40 (2000), pp. 443–451.

[381] T. VALKONEN, *Convergence of a SOR-Weiszfeld type algorithm for incomplete data sets*, Numerical Functional Analysis and Optimization, (2006). To appear.

[382] Y. VARDI AND C.-H. ZHANG, *The multivariate L1-median and associated data depth*, in Proceedings of the National Academy of Science, vol. 97, USA, February 2000, National Academy of Sciences, pp. 1423–1426.

[383] Y. VARDI AND C.-H. ZHANG, *A modified Weiszfeld algorithm for the Fermat-Weber location problem*, Mathematical Programming, 90 (2001), pp. 559–566.

[384] R. S. VARGA, *Matrix Iterative Analysis*, Prentice-Hall, Inc., 1962.

[385] P. VEHVILÄINEN, *Data Mining for Managing Intrinsic Quality of Service in Digital Mobile Telecommunications Networks*, PhD thesis, Institute of Automation and Control, Tampere University of Technology, Finland, 2004.

[386] J. VERBEEK, *Mixture models for clustering and dimension reduction*, PhD thesis, University of Amsterdam, 2004.

[387] B. S. VERKHOVSKY AND Y. S. POLYAKOV, *Feedback algorithm for the single-facility minisum problem*, Annals of the European Academy of Sciences, 1 (2003), pp. 127–136.

[388] P. A. VIJAYA, M. N. MURTY, AND D. K. SUBRAMANIAN, *Leaders-subleaders: an efficient hierarchical clustering algorithm for large data sets*, Pattern Recognition Letters, 25 (2004), pp. 505–513.

[389] S. VISURI, V. KOIVUNEN, AND H. OJA, *Sign and rank covariance matrices*, Journal of Statistical Planning and Inference, 91 (2000), pp. 557–575.

[390] S. VISURI, E. OLLILA, V. KOIVUNEN, J. MÖTTÖNEN, AND H. OJA, *Affine equivariant multivariate rank methods*, Journal of Statistical Planning and Inference, 114 (2003), pp. 161–185.

[391] M. S. VIVEROS, J. P. NEARHOS, AND M. J. ROTHMAN, *Applying data mining techniques to a health insurance information system*, in Proceedings of the 22nd International Conference on Very Large Data Bases, 1996, pp. 286–294.

[392] W. WANG, J. YANG, AND R. R. MUNTZ, *STING: A statistical information grid approach to spatial data mining*, in Proceedings of the 23nd International Conference on Very Large Databases (VLDB'97), 1997, pp. 186–195.

[393] J. WARD, *Hierarchical grouping to optimize an objective function*, Journal of the American Statistical Association, 58 (1963), pp. 236–244.

[394] G. WEISS, J. EDDY, AND S. WEISS, *Knowledge-based intelligent techniques in industry*, CRC Press, 1999, ch. Intelligent telecommunication technologies, pp. 249–276.

[395] E. W. WEISSTEIN, *www.mathworld.com*. World Wide Web.

[396] E. WEISZFELD, *Sur le point pour lequel les sommes des distances de n points donnés et minimum*, Tôhoku Mathematical Journal, 43 (1937), pp. 355–386.

[397] M. WOLFSON, Z. MADJD-SADJADI, AND P. JAMES, *Identifying National Types: A Cluster Analysis of Politics, Economics, and Conflict*, Journal of Peace Research, 41 (2004), pp. 607–623.

[398] M. WRIGHT, *Direct search methods: once scorned, now respectable*, in Proceedins of the 1995 Dundee Biennial Conference in Numerical Analysis, Harlow, United Kingdom, 1996, Addison Wesley Longman.

[399] J. WU, A. E. HASSAN, AND R. C. HOLT, *Comparison of clustering algorithms in the context of software evolution*, in Proceedings of ICSM 2005: International Conference on Software Maintenance, 2005.

[400] R. XU AND D. W. II, *Survey of clustering algorithms*, IEEE Transactions on Neural Networks, 16 (2005), pp. 645–678.

[401] X. XU, M. ESTER, H.-P. KRIEGEL, AND J. SANDER, *A distribution-based clustering algorithm for mining in large spatial databases*, in Proceedings of the 14th International Conference on Data Engineering (ICDE 98), 1998, pp. 324–331.

[402] X. XU, J. JÄGER, AND H.-P. KRIEGEL, *A fast parallel clustering algorithm for large spatial databases*, Data Mining and Knowlegde Discovery, 3 (1999), pp. 263–290.

[403] Y. YANG, X. GUAN, AND J. YOU, *CLOPE: a fast and effective clustering algorithm for transactional data*, in Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 682–687.

[404] J. YATES, W. ORLIKOWSKI, AND K. OKAMURA, *Explicit and implicit structuring of genres in electronic communication: Reinforcement and change of social interaction*, Organization Science, 10 (1999), pp. 83–103.

[405] J. YATES, W. J. ORLIKOWSKI, AND J. RENNECKER, *Collaborative genres for collaboration: Genre systems in digital media*, in Proceedings of the 30th Annual Hawaii International Conference on System Sciences (HICSS-30), vol. 6, IEEE Computer Society, 1997, pp. 50–59.

[406] K. Y. YEUNG, R. E. BUMGARNER, AND A. E. RAFTERY, *Bayesian model averaging: development of an improved multi-class, gene selection and classification tool for microarray data*, Bioinformatics, 21 (2005), pp. 2394–2402.

[407] K. Y. YEUNG, C. FRALEY, A. MURUA, A. E. RAFTERY, AND W. L. RUZZO, *Model-based clustering and data transformations for gene expression data*, Bioinformatics, 17 (2001), pp. 977–987.

[408] K. Y. YEUNG, D. R. HAYNOR, AND W. L. RUZZO, *Validating clustering for gene expression data*, Bioinformatics, 17 (2001), pp. 309–318.

[409] K. Y. YEUNG AND W. L. RUZZO, *Principal component analysis for clustering gene expression data*, Bioinformatics, 17 (2001), pp. 763–774.

[410] V. J. YOHAI AND R. H. ZAMAR, *High breakdown-point estimates of regression by means of the minimization of an efficient scale*, Journal of the American Statistical Association, 83 (1988), pp. 406–413.

[411] T. YOSHIOKA AND G. HERMAN, *Coordinating information using genres*, tech. report, Massachusetts Institute of Technology, Center for Coordination Science, August 2000.

[412] K. YOSIDA, *Functional analysis*, Springer-Verlag, Berlin Heidelberg New York, 1974.

[413] B. ZHANG, *Generalized K-harmonic means – boosting in unsupervised learning*, Tech. Report 137, Hewlett Packard, October 2000.

[414] B. ZHANG AND S. N. SRIHARI, *Binary vector dissimilarity measures for handwriting identification*, in Proceedings of SPIE, Document Recognition and Retrieval X, Santa Clara, California, USA, January 2003, pp. 155–166.

[415] ——, *Properties of binary vector dissimilarity measures*, in Proceedings of the Seventh Joint Conference on Information Sciences - Computer Vision, Pattern recognition and Image processing, 2003.

[416] D.-Q. ZHANG AND S.-C. CHEN, *Clustering incomplete data using kernel-based fuzzy c-means algorithm*, Neural Processing Letters, 18 (2003), pp. 155–162.

[417] J. ZHANG AND G. LI, *Breakdown properties of location M-estimators*, The Annals of Statistics, 26 (1998), pp. 1170–1189.

[418] T. ZHANG, R. RAMAKRISHNAN, AND M. LIVNY, *Birch: an efficient data clustering method for very large databases*, in SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data, New York, NY, USA, 1996, ACM Press, pp. 103–114.

[419] ——, *Birch: A new data clustering algorithm and its applications*, Data Mining and Knowledge Discovery, 1 (1997), pp. 141–182.

[420] Z. ZHANG, *Parameter estimation techniques: A tutorial with application to conic fitting*, Image and Vision Computing Journal, 15 (1997), pp. 59–76.

[421] S. ZHONG, T. M. KHOSHGOFTAAR, AND N. SELIYA, *Analyzing software measurement data with clustering techniques*, IEEE Intelligent Systems, 19 (2004), pp. 20–27.

[422] ——, *Unsupervised learning for expert-based software quality estimation*, in Proceedings of the Eighth IEEE International Symposium on High Assurance Systems Engineering (HASE04), Tampa, FL, USA, March 2004, IEEE Computer Society, pp. 149–155.

[423] D. ZHU, A. PORTER, S. CUNNINGHAM, J. CARLISIE, AND A. NAYAK, *A process for mining science and technology documents databases, illustrated for the case of "knowledge discovery and data mining"*, Ciência da Informacão, 28 (1999), pp. 7–14.

[424] Y. ZUO AND R. SERFLING, *On the performance of some robust nonparametric location measures relative to a general notion of multivariate symmetry*, Journal of Statistical Planning and Inference, 84 (2000), pp. 55–79.

APPENDIX 1 NUMERICAL RESULTS ON SPATIAL MEDIAN ALGORITHMS

The following notation are used in the tables:

CG	conjugate gradient method
GS	golden section method
NM	Nelder-Mead method
SOR	the spatial median by successive overrelaxation method
ASSOR	the spatial median by successive overrelaxation method using active sets
MW	modified Weiszfeld
CG1NM	CG initialized NM method
p	number of dimensions
d	index of data set
n	number of data points
$\mathcal{J}^*$	value of cost function at the optimum
$(\mathbf{u}^*)_1$	value of the first vector component at optimum
$(\mathbf{u}^*)_2$	value of the second vector component at optimum
$(\mathbf{g}^*)_1$	value of the first component of gradient vector at optimum
$(\mathbf{g}^*)_2$	value of the second component of gradient vector at optimum
#CGi	number of iterations by CG method
#GSi	number of iterations by GS method
#NMi	number of iterations by NM method
#NM	number of function evaluations by NM method
#CG	number of function evaluations by CG method
#fev	total number of function evaluations
it	total number of iterations
$e(\mathcal{J}(\mathbf{u}^*))$	difference between the solution and reference value
$e(\mathbf{u}^*)$	location error to reference solution
total	total number of function evaluations
sp	index of initial clusters
cw-med	coordinate-wise median

TABLE 15 Reference solutions of the spatial median problem on the test data sets. Solved with CG1NM by starting from the mean of a data set and terminating according to the following stopping criteria: CG:10^{-1}, GS:10^{-6} and NM:10^{-12}.

Reference solutions													
p	data	n	$\mathcal{J}^*$	$(\mathbf{u}^*)_1$	$(\mathbf{u}^*)_2$	$(\mathbf{g}^*)_1$	$(\mathbf{g}^*)_2$	#CGi	#GSi	#CG	#NMi	#NM	#fev
2	1	5	5.6568542495	+0.00e+00	+0.00e+00	+0.00e+00	+0.00e+00	1	0	4	58	117	121
	2	20	12.4605534804	-4.76e-02	-4.79e-02	-7.61e-01	-2.61e-02	1	29	62	98	186	248
	3	100	52.2666419878	-3.25e-02	-2.43e-02	+4.42e-07	+4.69e-07	1	29	62	68	154	216
	4	500	257.1516146174	-1.09e-03	-1.66e-02	+1.65e-06	+5.06e-06	1	29	62	55	132	194
	5	40	28.0124414406	+4.34e-03	+2.10e-03	+9.63e-08	+4.17e-08	1	29	62	59	137	199
	6	100	83.2955042800	+1.75e-02	+1.46e-02	-1.43e-06	-9.09e-07	1	29	62	61	143	205
	7	300	29.7792096461	-8.85e-04	-1.64e-04	+3.52e-06	+1.00e-05	1	29	62	61	134	196
	8	100	117.1020022524	+9.66e-02	+2.11e-02	-1.96e-06	+6.66e-07	2	58	121	61	140	261
8	1	100	166.5910085601					1	29	62	359	690	752
	2	300	59.5584192922					1	29	62	600	964	1026
16	1	100	235.5952636751					1	29	62	903	1530	1592
	2	300	84.2283243165					1	29	62	2708	3730	3792
32	1	100	333.1820171202					1	29	62	2136	3387	3449
	2	300	119.1168385844					1	29	62	11261	13776	13838
64	1	100	471.1905273501					1	29	62	11107	14492	14554
	2	300	168.4566486331					1	29	62	58780	65696	65758

TABLE 16 Results of NM and CG1NM methods on the bivariate data sets.

2D NM and CG1NM									
		NM			CG1NM				
data	sp	$e(\mathcal{J}(\mathbf{u}^*))$	total	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	#CG	#NM	total	$e(\mathbf{u}^*)$
1	1	3.49e-07	99	3.37e-07	5.37e-07	96	54	150	5.01e-07
n=5	2	2.88e-07	111	2.82e-07	4.91e-07	96	55	151	4.83e-07
	3	3.22e-07	113	2.80e-07	3.45e-07	127	61	188	2.96e-07
	4	0.00e+00	37	0.00e+00	0.00e+00	4	37	41	0.00e+00
2	1	4.93e-08	129	1.52e-07	3.42e-07	127	67	194	1.28e-06
n=20	2	2.07e-07	127	5.80e-07	1.35e-07	127	76	203	2.68e-07
	3	6.72e-08	132	2.81e-07	3.09e-07	127	70	197	1.29e-06
	4	1.24e-07	107	4.96e-07	2.20e-07	65	70	135	5.82e-07
3	1	4.97e-12	96	2.17e-07	8.90e-12	96	52	148	2.99e-07
n=100	2	1.13e-11	110	3.81e-07	5.78e-12	96	49	145	2.44e-07
	3	9.46e-12	111	2.55e-07	1.63e-11	96	49	145	4.44e-07
	4	7.11e-12	95	2.82e-07	5.21e-12	65	49	114	2.26e-07
4	1	1.28e-11	100	1.48e-07	7.24e-11	127	38	165	4.41e-07
n=500	2	1.19e-10	106	5.14e-07	4.05e-11	96	38	134	3.25e-07
	3	3.67e-11	115	2.64e-07	5.38e-11	127	38	165	3.93e-07
	4	2.50e-11	87	2.05e-07	2.22e-11	65	53	118	2.33e-07
5	1	7.39e-13	114	1.57e-07	8.85e-13	127	55	182	2.49e-07
n=40	2	7.14e-13	115	2.85e-07	2.07e-12	96	51	147	3.58e-07
	3	1.28e-12	105	3.75e-07	5.51e-13	127	71	198	2.09e-07
	4	4.35e-04	39	4.34e-03	4.76e-13	34	52	86	2.21e-07
6	1	4.59e-12	92	2.87e-07	8.16e-12	65	46	111	3.90e-07
n=100	2	1.98e-12	117	1.68e-07	8.95e-12	158	46	204	4.26e-07
	3	4.15e-12	112	2.04e-07	4.22e-12	96	45	141	2.53e-07
	4	1.67e-11	86	5.68e-07	3.89e-12	65	45	110	2.28e-07
7	1	5.79e-10	97	2.02e-07	8.16e-10	65	43	108	2.40e-07
n=300	2	1.44e-09	115	3.64e-07	1.14e-09	96	41	137	2.93e-07
	3	6.09e-11	119	8.16e-08	2.36e-09	96	47	143	4.67e-07
	4	1.15e-09	77	3.61e-07	9.70e-10	34	44	78	3.64e-07
8	1	7.18e-12	101	2.93e-07	1.04e-11	96	50	146	3.88e-07
n=100	2	2.47e-12	127	2.01e-07	4.65e-12	158	53	211	2.59e-07
	3	4.96e-12	111	3.38e-07	2.87e-12	96	54	150	2.46e-07
	4	1.90e-12	122	1.52e-07	2.30e-12	96	55	151	2.06e-07

TABLE 17 Results of CG1 and CG2 methods on the bivariate data sets.

2D CG1 and CG2							
		CG1			CG2		
data	sp	$e(\mathcal{J}(\mathbf{u}^*))$	total	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	total	$e(\mathbf{u}^*)$
1	1	4.36e-09	240	3.08e-09	4.36e-09	240	3.08e-09
n=5	2	4.06e-09	319	3.90e-09	3.03e-09	398	2.98e-09
	3	2.28e-09	398	1.97e-09	4.13e-09	477	4.13e-09
	4	0.00e+00	4	0.00e+00	0.00e+00	4	0.00e+00
2	1	4.23e-07	1583	1.63e-06	3.00e-09	477	4.58e-09
n=20	2	2.08e-07	793	8.70e-07	3.38e-09	556	8.93e-09
	3	2.21e-07	1504	8.93e-07	2.98e-09	477	1.19e-08
	4	1.96e-07	1030	8.12e-07	3.24e-09	398	1.36e-08
3	1	8.53e-14	398	2.50e-08	5.72e-02	556	2.52e-02
n=100	2	9.24e-14	477	2.94e-08	8.61e-04	240	3.24e-03
	3	0.00e+00	477	6.23e-09	6.03e-02	1346	2.60e-02
	4	1.49e-13	398	3.14e-08	4.93e-02	240	2.44e-02
4	1	3.98e-13	556	2.07e-08	2.93e-01	398	2.46e-02
n=500	2	1.71e-13	398	1.24e-08	5.15e-01	161	3.46e-02
	3	1.71e-13	477	9.83e-09	6.74e-02	319	1.30e-02
	4	1.71e-13	319	1.07e-08	1.81e-04	398	5.84e-04
5	1	9.59e-14	793	1.05e-07	1.23e-03	240	1.22e-02
n=40	2	5.68e-14	714	7.41e-08	1.16e-01	161	1.17e-01
	3	1.33e-12	951	3.89e-07	5.35e-01	240	1.92e-01
	4	1.07e-14	398	3.82e-08	1.80e-08	240	4.29e-05
6	1	2.84e-14	477	5.52e-09	4.82e-03	398	7.31e-03
n=100	2	4.26e-14	635	2.55e-08	1.34e+00	319	1.20e-01
	3	1.42e-14	477	1.39e-08	4.77e-03	240	8.96e-03
	4	1.14e-13	398	5.07e-08	7.24e-02	161	2.82e-02
7	1	1.44e-10	556	1.11e-07	2.84e-04	319	1.94e-04
n=300	2	9.83e-12	477	3.69e-08	3.45e-04	240	2.19e-04
	3	1.68e-11	556	4.47e-08	2.75e-04	319	1.94e-04
	4	1.73e-10	398	9.71e-08	1.25e-03	240	4.22e-04
8	1	3.13e-13	635	9.79e-08	2.05e+00	161	2.29e-01
n=100	2	2.98e-13	635	6.15e-08	8.67e-02	398	4.78e-02
	3	2.84e-14	556	2.68e-08	4.12e-01	319	1.01e-01
	4	7.82e-13	635	1.22e-07	3.66e-02	240	3.20e-02

TABLE 18 Results of Modified Weiszfeld, SOR, and ASSOR methods on the bivariate data sets.

2D Modified Weiszfeld, SOR, and ASSOR										
		MW			SOR			ASSOR		
data	sp	$e(\mathcal{J}(\mathbf{u}^*))$	it	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	it	$e(\mathbf{u}^*)$	$e(\mathcal{J}(\mathbf{u}^*))$	it	$e(\mathbf{u}^*)$
1	1	2.86e-10	6	2.02e-10	4.34e-07	23	3.07e-07	4.57e-07	18	3.23e-07
n=5	2	1.69e-14	6	1.44e-14	2.35e-07	19	2.24e-07	2.36e-07	19	2.25e-07
	3	1.84e-11	7	1.37e-11	2.41e-07	19	2.11e-07	2.42e-07	19	2.12e-07
	4	0.00e+00	1	0.00e+00	0.00e+00	1	0.00e+00	0.00e+00	1	0.00e+00
2	1	7.25e-07	33	3.04e-06	3.41e-05	20	1.42e-04	2.95e-07	23	1.24e-06
n=20	2	6.42e-07	34	2.69e-06	3.41e-05	17	1.42e-04	3.04e-07	22	1.27e-06
	3	7.36e-07	34	3.08e-06	3.42e-05	17	1.42e-04	2.81e-07	23	1.18e-06
	4	7.01e-07	30	2.94e-06	3.42e-05	15	1.42e-04	2.88e-07	21	1.21e-06
3	1	1.23e-10	19	1.01e-06	6.30e-12	13	2.11e-07	2.29e-11	10	3.92e-07
n=100	2	6.12e-11	20	7.35e-07	1.47e-11	11	3.86e-07	1.77e-11	11	4.06e-07
	3	1.36e-10	19	1.08e-06	1.35e-11	11	3.15e-07	1.07e-11	11	2.95e-07
	4	1.66e-10	16	1.15e-06	3.73e-12	10	1.57e-07	2.34e-12	10	1.37e-07
4	1	1.63e-10	20	6.48e-07	1.55e-11	12	2.05e-07	2.22e-11	11	2.45e-07
n=500	2	3.70e-10	17	9.35e-07	1.30e-11	11	1.72e-07	1.69e-11	11	1.94e-07
	3	2.40e-10	18	8.10e-07	2.04e-11	10	2.25e-07	2.45e-11	10	2.46e-07
	4	4.52e-10	16	1.07e-06	1.00e-11	10	1.51e-07	1.30e-11	10	1.72e-07
5	1	4.25e-11	36	2.26e-06	7.13e-12	19	9.25e-07	1.11e-11	24	1.16e-06
n=40	2	4.09e-11	34	2.22e-06	1.85e-11	8	1.49e-06	1.85e-11	8	1.49e-06
	3	3.76e-11	41	2.13e-06	1.43e-11	26	1.31e-06	1.44e-11	26	1.31e-06
	4	5.78e-11	19	2.64e-06	1.34e-11	13	1.28e-06	1.34e-11	13	1.28e-06
6	1	2.50e-11	19	6.91e-07	1.65e-12	12	1.84e-07	5.64e-12	9	2.85e-07
n=100	2	2.63e-11	21	7.03e-07	2.22e-12	12	1.83e-07	1.31e-12	12	1.40e-07
	3	2.08e-11	18	5.78e-07	2.56e-12	12	2.01e-07	1.59e-12	12	1.58e-07
	4	6.16e-11	16	9.84e-07	8.47e-12	9	3.90e-07	1.07e-11	9	4.33e-07
7	1	1.14e-08	19	1.26e-06	9.05e-08	14	2.52e-06	3.90e-09	12	7.39e-07
n=300	2	1.62e-08	16	1.50e-06	8.55e-08	12	2.65e-06	3.00e-09	12	6.48e-07
	3	1.47e-08	19	1.43e-06	9.03e-08	13	2.52e-06	2.27e-09	13	5.64e-07
	4	9.85e-09	18	1.17e-06	8.61e-08	7	2.68e-06	1.23e-09	8	4.15e-07
8	1	4.77e-11	26	1.16e-06	1.59e-11	17	6.84e-07	4.82e-12	16	3.88e-07
n=100	2	7.02e-11	29	1.40e-06	1.33e-11	17	6.26e-07	1.28e-11	17	6.15e-07
	3	6.56e-11	22	1.30e-06	2.17e-11	16	7.33e-07	2.23e-11	16	7.45e-07
	4	7.55e-11	25	1.40e-06	4.12e-12	16	3.03e-07	4.43e-12	16	3.15e-07

TABLE 19 Results of NM and CG1NM methods on the multidimensional data sets. (*The algorithm exceeded the maximum number of function evaluations.)

			NM			CG1NM				
p	data	sp	$e(\mathcal{J}^*)$	#fev	$e(\mathbf{u}^*)$	$e(\mathcal{J}^*)$	#CG	#NM	#fev	$e(\mathbf{u}^*)$
8	1	1	2.96e-11	879	6.31e-07	1.87e-11	96	214	310	4.26e-07
	n=100	2	2.13e-02	2172	1.71e-02	9.51e-11	220	234	454	9.03e-07
		3	1.49e-10	1013	1.09e-06	2.82e-11	189	265	454	5.55e-07
		4	4.34e-11	888	5.43e-07	1.65e-11	65	261	326	3.90e-07
	2	1	6.82e-09	995	5.50e-07	1.69e-08	96	282	378	9.14e-07
	n=300	2	5.11e-01	1346	8.73e-03	4.64e-09	127	325	452	5.76e-07
		3	4.75e-09	1021	6.49e-07	5.74e-09	158	597	755	4.87e-07
		4	2.29e-09	755	2.89e-07	4.11e-09	34	384	418	3.69e-07
16	1	1	3.23e-10	4793	1.77e-06	2.19e-11	96	489	585	4.55e-07
	n=100	2	2.61e-02	7264	1.76e-02	3.26e-11	251	454	705	5.82e-07
		3	1.61e-10	9094	1.01e-06	4.74e-11	220	462	682	6.67e-07
		4	2.68e-10	4406	1.52e-06	2.51e-11	65	491	556	3.67e-07
	2	1	5.66e-08	5705	1.50e-06	4.87e-08	96	788	884	1.29e-06
	n=300	2	6.18e-01	3724	1.13e-02	2.65e-08	158	1187	1345	1.36e-06
		3	8.72e-09	7929	5.24e-07	1.50e-08	220	695	915	8.76e-07
		4	1.12e-08	5648	6.24e-07	5.45e-08	34	2132	2166	1.25e-06
32	1	1	1.66e-06	55961	1.30e-04	3.40e-10	127	1906	2033	1.49e-06
	n=100	2	4.74e-02	22684	2.25e-02	4.88e-11	344	1187	1531	7.19e-07
		3	5.82e+00	100000*	1.98e-01	3.57e-10	282	1586	1868	2.05e-06
		4	4.36e-06	47538	1.61e-04	5.12e-11	65	1613	1678	6.86e-07
	2	1	9.90e-03	100000*	7.06e-04	9.18e-09	127	1220	1347	6.25e-07
	n=300	2	1.39e+00	29798	1.19e-02	1.31e-07	220	5545	5765	2.04e-06
		3	4.25e+01	100000*	1.13e-01	7.55e-08	282	2420	2702	1.40e-06
		4	2.66e-02	97048	1.34e-03	2.51e-07	34	6339	6373	3.44e-06
64	1	1	3.74e-05	100001*	4.58e-04	6.95e-10	158	5050	5208	3.06e-06
	n=100	2	7.34e-02	100000*	2.38e-02	7.13e-11	437	3779	4216	7.57e-07
		3	1.03e+00	100000*	1.23e-01	2.06e-11	375	2756	3131	5.64e-07
		4	5.69e-05	100000*	5.71e-04	1.88e-10	65	4903	4968	1.09e-06
	2	1	4.76e-02	100000*	1.35e-03	1.84e-07	158	6034	6192	2.61e-06
	n=300	2	3.25e+00	100000*	2.64e-02	7.83e-08	282	4757	5039	1.91e-06
		3	2.29e+01	100000*	6.57e-02	1.91e-07	375	11878	12253	3.53e-06
		4	2.24e-04	100000*	7.76e-05	7.84e-07	34	18594	18628	7.42e-06

TABLE 20 Results of CG1 and CG2 methods on the multidimensional data sets (*The algorithm exceeded the maximum number of function evaluations.)

			CG1			CG2		
p	data	sp	$e(\mathcal{J}^*)$	#fev	$e(\mathbf{u}^*)$	$e(\mathcal{J}^*)$	#fev	$e(\mathbf{u}^*)$
8	1	1	3.13e-13	477	5.35e-08	3.36e+00	161	1.52e-01
	n=100	2	8.53e-14	793	4.33e-08	3.43e-02	556	1.81e-02
		3	0.00e+00	556	2.78e-08	3.68e-02	556	1.84e-02
		4	1.99e-13	398	5.85e-08	1.45e-01	161	2.82e-02
	2	1	1.27e-09	477	2.31e-07	1.12e-03	319	2.75e-04
	n=300	2	1.68e-11	556	3.56e-08	3.46e-08	319	1.54e-06
		3	6.11e-11	793	4.15e-08	4.14e-05	714	5.39e-05
		4	3.47e-10	398	9.64e-08	2.51e-03	240	4.22e-04
16	1	1	1.71e-13	477	6.53e-08	1.32e-01	240	2.41e-02
	n=100	2	-2.27e-13	872	4.72e-08	6.78e+00	635	1.91e-01
		3	-2.27e-13	793	5.22e-08	1.29e-02	872	7.14e-03
		4	0.00e+00	398	9.73e-08	2.05e-01	161	2.82e-02
	2	1	1.21e-09	477	2.33e-07	1.51e-03	319	1.98e-04
	n=300	2	2.65e-11	635	3.84e-08	1.37e-03	556	2.40e-04
		3	4.31e-11	872	4.61e-08	7.85e-06	872	1.96e-05
		4	4.88e-10	398	9.76e-08	3.55e-03	240	4.22e-04
32	1	1	-8.53e-13	635	8.58e-08	6.62e+00	240	1.33e-01
	n=100	2	-6.82e-13	1109	8.92e-08	1.78e-01	872	2.88e-02
		3	-1.71e-13	951	1.11e-07	4.26e-03	1030	3.76e-03
		4	-2.27e-13	398	9.83e-08	2.90e-01	161	2.82e-02
	2	1	2.54e-09	556	2.45e-07	1.90e-03	398	2.51e-04
	n=300	2	9.09e-12	714	3.81e-08	4.63e-03	793	3.94e-04
		3	3.24e-11	1030	4.31e-08	1.09e-03	872	1.67e-04
		4	6.90e-10	398	1.15e-07	5.02e-03	240	4.22e-04
64	1	1	-1.19e-12	635	1.70e-07	7.74e+00	398	1.25e-01
	n=100	2	-1.93e-12	1346	1.35e-07	3.65e-02	1346	8.21e-03
		3	-1.99e-12	1188	1.56e-07	3.85e-03	1030	3.38e-03
		4	-1.42e-12	398	1.32e-07	4.10e-01	161	2.82e-02
	2	1	-8.83e-11	714	7.73e-08	1.56e-02	556	5.98e-04
	n=300	2	3.92e-11	872	8.24e-08	1.87e-03	872	2.05e-04
		3	-2.11e-11	1267	6.99e-08	2.35e-04	1267	7.64e-05
		4	8.72e-10	398	1.61e-07	7.09e-03	240	4.22e-04

TABLE 21 Results of Modified Weiszfeld, SOR, and ASSOR methods on the multidimensional data sets.

			MW			SOR			ASSOR		
p	data	sp	$e(\mathcal{J}*)$	#it	$e(\mathbf{u}^*)$	$e(\mathcal{J}*)$	#it	$e(\mathbf{u}^*)$	$e(\mathcal{J}^*)$	#it	$e(\mathbf{u}^*)$
8	1	1	5.00e-11	19	6.99e-07	2.59e-12	12	1.76e-07	1.13e-11	9	3.00e-07
	n=100	2	5.26e-11	21	7.11e-07	3.04e-12	12	1.66e-07	2.61e-12	12	1.55e-07
		3	4.15e-11	18	5.87e-07	3.61e-12	12	1.84e-07	3.15e-12	12	1.73e-07
		4	1.23e-10	16	9.92e-07	2.00e-11	9	4.30e-07	2.13e-11	9	4.41e-07
	2	1	2.28e-08	19	1.26e-06	2.31e-08	13	9.99e-07	7.81e-09	12	7.38e-07
	n=300	2	3.23e-08	16	1.50e-06	1.40e-08	12	7.45e-07	6.00e-09	12	6.47e-07
		3	2.94e-08	19	1.44e-06	1.82e-08	13	7.83e-07	4.54e-09	13	5.65e-07
		4	1.97e-08	18	1.17e-06	1.21e-08	8	7.15e-07	2.46e-09	8	4.14e-07
16	1	1	7.05e-11	19	7.36e-07	3.75e-12	12	2.16e-07	1.56e-11	9	3.04e-07
	n=100	2	7.42e-11	21	7.48e-07	3.64e-12	12	1.64e-07	3.27e-12	12	1.59e-07
		3	5.83e-11	18	5.81e-07	4.52e-12	12	1.82e-07	4.09e-12	12	1.77e-07
		4	1.74e-10	16	1.03e-06	2.90e-11	9	4.73e-07	2.98e-11	9	4.78e-07
	2	1	3.22e-08	19	1.26e-06	1.10e-08	15	6.36e-07	1.10e-08	12	7.42e-07
	n=300	2	4.57e-08	16	1.51e-06	9.68e-09	12	4.80e-07	8.49e-09	12	6.51e-07
		3	4.16e-08	19	1.44e-06	1.21e-08	13	6.79e-07	6.43e-09	13	5.66e-07
		4	2.78e-08	18	1.18e-06	5.96e-09	8	3.89e-07	3.48e-09	8	4.17e-07
32	1	1	9.92e-11	19	7.35e-07	4.83e-12	12	2.18e-07	2.19e-11	9	3.71e-07
	n=100	2	1.05e-10	21	7.47e-07	4.72e-12	12	2.29e-07	4.43e-12	12	2.26e-07
		3	8.23e-11	18	6.32e-07	5.80e-12	12	2.47e-07	5.57e-12	12	2.44e-07
		4	2.46e-10	16	1.03e-06	4.14e-11	9	4.75e-07	4.19e-11	9	4.78e-07
	2	1	4.56e-08	19	1.28e-06	1.21e-08	12	6.49e-07	1.56e-08	12	7.47e-07
	n=300	2	6.47e-08	16	1.52e-06	1.12e-08	12	5.69e-07	1.20e-08	12	6.56e-07
		3	5.89e-08	19	1.45e-06	1.17e-08	13	6.36e-07	9.08e-09	13	5.78e-07
		4	3.94e-08	18	1.19e-06	5.09e-09	8	3.41e-07	4.91e-09	8	4.23e-07
64	1	1	1.39e-10	19	7.63e-07	8.19e-12	12	2.67e-07	2.98e-11	9	3.86e-07
	n=100	2	1.47e-10	21	7.75e-07	5.46e-12	12	2.42e-07	5.12e-12	12	2.41e-07
		3	1.15e-10	18	7.04e-07	6.93e-12	12	2.60e-07	6.76e-12	12	2.59e-07
		4	3.47e-10	16	1.06e-06	5.78e-11	9	5.04e-07	5.82e-11	9	5.05e-07
	2	1	6.43e-08	19	1.34e-06	1.61e-08	13	6.40e-07	2.20e-08	12	7.67e-07
	n=300	2	9.14e-08	16	1.58e-06	1.57e-08	12	6.32e-07	1.69e-08	12	6.76e-07
		3	8.32e-08	19	1.51e-06	1.41e-08	13	6.68e-07	1.27e-08	13	6.39e-07
		4	5.56e-08	18	1.25e-06	6.41e-09	8	4.02e-07	6.85e-09	8	4.43e-07

TABLE 22 Normal: Relative efficiency of the coordinate-wise median, SOR, and ASSOR spatial median estimators with respect to the sample mean. Laplace: Relative efficiency of the sample mean, SOR, and ASSOR spatial median estimators with respect to the coordinate-wise median.

Relative efficiency (avg. of p variance estimates over 100 samples)							
Normal distribution ($N_p(\mathbf{0}, \mathbf{1}_p)$)							
%(missing)	p	2	4	8	16	32	64
0	cw-med	0.642	0.615	0.626	0.639	0.642	0.635
	SOR	0.796	0.864	0.928	0.972	0.985	0.992
	ASSOR	0.796	0.864	0.928	0.972	0.985	0.992
10	cw-med	0.638	0.650	0.647	0.646	0.632	0.639
	SOR	0.770	0.876	0.923	0.969	0.983	0.991
	ASSOR	0.770	0.876	0.923	0.969	0.983	0.991
20	cw-med	0.654	0.640	0.641	0.641	0.643	0.645
	SOR	0.745	0.825	0.924	0.957	0.980	0.988
	ASSOR	0.745	0.825	0.924	0.957	0.980	0.988
30	cw-med	0.666	0.635	0.625	0.644	0.644	0.648
	SOR	0.718	0.801	0.899	0.951	0.976	0.987
	ASSOR	0.718	0.801	0.899	0.951	0.976	0.987
40	cw-med	0.662	0.639	0.637	0.644	0.645	0.646
	SOR	0.712	0.769	0.877	0.934	0.970	0.983
	ASSOR	0.712	0.769	0.877	0.934	0.970	0.983
50	cw-med	0.619	0.638	0.648	0.640	0.640	0.640
	SOR	0.663	0.735	0.838	0.922	0.960	0.979
	ASSOR	0.663	0.734	0.838	0.922	0.960	0.979
Laplace distribution($L_p(\mathbf{0}, \mathbf{1}_p)$)							
0	mean	0.579	0.564	0.539	0.558	0.560	0.560
	SOR	0.864	0.738	0.635	0.606	0.590	0.574
	ASSOR	0.864	0.738	0.635	0.606	0.590	0.574
10	mean	0.554	0.558	0.565	0.552	0.562	0.563
	SOR	0.866	0.745	0.669	0.609	0.591	0.580
	ASSOR	0.866	0.745	0.669	0.609	0.591	0.580
20	mean	0.572	0.552	0.570	0.556	0.566	0.566
	SOR	0.860	0.747	0.685	0.622	0.599	0.583
	ASSOR	0.860	0.747	0.685	0.622	0.599	0.583
30	mean	0.570	0.577	0.552	0.565	0.561	0.568
	SOR	0.886	0.781	0.656	0.633	0.596	0.586
	ASSOR	0.886	0.781	0.656	0.633	0.596	0.586
40	mean	0.577	0.565	0.566	0.573	0.581	0.581
	SOR	0.900	0.815	0.709	0.649	0.623	0.603
	ASSOR	0.900	0.815	0.709	0.649	0.623	0.603
50	mean	0.571	0.583	0.597	0.586	0.577	0.584
	SOR	0.893	0.824	0.740	0.675	0.627	0.609
	ASSOR	0.893	0.824	0.740	0.675	0.627	0.609

TABLE 23 Consistency of the estimators in the presence of missing data.

Statistical consistency (avg. bias over 100 sample)									
$N_p(\mathbf{0}, \mathbf{1}_p)$									
		2D				8D			
%(missing)	n	mean	cw-med	SOR	ASSOR	mean	cw-med	SOR	ASSOR
0	10	3.9e-2	6.9e-2	5.2e-2	5.2e-2	1.0e-1	1.0e-1	1.0e-1	1.0e-1
	10^2	9.3e-3	5.6e-3	1.0e-2	1.0e-2	2.3e-2	2.6e-2	2.2e-2	2.2e-2
	10^3	4.5e-3	4.3e-3	4.1e-3	4.1e-3	9.0e-3	1.3e-2	9.6e-3	9.6e-3
	10^4	2.1e-3	2.2e-3	2.4e-3	2.4e-3	3.5e-3	3.9e-3	3.4e-3	3.4e-3
	10^5	4.2e-4	6.8e-4	3.9e-4	3.9e-4	7.8e-4	8.0e-4	7.5e-4	7.5e-4
15	10	1.9e-2	8.3e-3	2.2e-2	2.2e-2	6.1e-2	1.0e-1	5.6e-2	5.6e-2
	10^2	8.2e-3	1.9e-3	2.1e-2	2.1e-2	3.6e-2	3.8e-2	3.7e-2	3.7e-2
	10^3	8.5e-3	5.4e-3	8.3e-3	8.3e-3	9.4e-3	8.6e-3	1.0e-2	1.0e-2
	10^4	2.8e-4	6.5e-4	7.9e-4	7.9e-4	4.7e-3	5.5e-3	5.1e-3	5.1e-3
	10^5	4.5e-4	9.2e-4	9.8e-4	9.8e-4	1.1e-3	1.2e-3	9.8e-4	9.8e-4
40	10	5.7e-2	6.3e-2	6.1e-2	6.1e-2	1.1e-1	1.3e-1	1.2e-1	1.2e-1
	10^2	2.2e-2	2.0e-2	1.6e-2	1.6e-2	3.2e-2	4.1e-2	2.8e-2	2.8e-2
	10^3	2.8e-3	2.9e-3	3.3e-3	3.3e-3	1.1e-2	1.4e-2	1.2e-2	1.2e-2
	10^4	3.4e-3	3.7e-3	3.0e-3	3.0e-3	3.4e-3	5.4e-3	4.2e-3	4.2e-3
	10^5	7.7e-4	5.7e-4	4.7e-4	4.7e-4	8.8e-4	1.2e-3	1.4e-3	1.4e-3
$L_p(\mathbf{0}, \mathbf{1}_p)$									
		2D				8D			
% (missing)	n	mean	cw-med	SOR	ASSOR	mean	cw-med	SOR	ASSOR
0	10	3.9e-2	7.3e-3	3.0e-2	3.0e-2	1.3e-1	1.0e-1	1.1e-1	1.1e-1
	10^2	8.2e-3	3.7e-3	5.0e-3	5.0e-3	3.0e-2	1.9e-2	2.8e-2	2.8e-2
	10^3	2.8e-3	3.8e-3	2.0e-3	2.0e-3	1.0e-2	7.3e-3	8.1e-3	8.1e-3
	10^4	6.6e-4	8.9e-4	1.1e-3	1.1e-3	3.2e-3	2.7e-3	3.1e-3	3.1e-3
	10^5	5.3e-4	4.3e-4	5.3e-4	5.3e-4	1.1e-3	7.9e-4	1.2e-3	1.2e-3
15	10	1.2e-1	7.2e-2	6.8e-2	6.8e-2	1.4e-1	1.6e-1	1.3e-1	1.3e-1
	10^2	2.9e-2	9.2e-3	9.7e-3	9.7e-3	4.0e-2	2.7e-2	3.4e-2	3.4e-2
	10^3	2.7e-3	3.2e-3	5.9e-3	5.9e-3	1.8e-2	1.3e-2	1.6e-2	1.6e-2
	10^4	1.7e-3	1.1e-3	1.1e-3	1.1e-3	5.1e-3	2.8e-3	4.9e-3	4.9e-3
	10^5	5.3e-4	6.1e-4	3.2e-4	3.2e-4	1.5e-3	1.0e-3	1.8e-3	1.8e-3
40	10	5.2e-2	2.2e-2	6.6e-2	6.6e-2	1.6e-1	1.3e-1	1.5e-1	1.5e-1
	10^2	1.5e-2	1.0e-2	1.3e-2	1.3e-2	5.5e-2	3.8e-2	3.7e-2	3.7e-2
	10^3	4.7e-3	4.2e-3	3.3e-3	3.2e-3	2.2e-2	1.4e-2	1.8e-2	1.8e-2
	10^4	2.1e-3	7.3e-4	1.3e-3	1.3e-3	4.2e-3	3.4e-3	3.6e-3	3.6e-3
	10^5	6.0e-4	3.8e-4	4.6e-4	4.5e-4	1.9e-3	1.0e-3	1.6e-3	1.6e-3

APPENDIX 2 SOR RELAXATION PARAMETER VALUE

The following figures depict the maximum, minimum and mean effect of relaxation parameter ω to the number of SOR iterations on the synthetic test data sets (the curves are averages over 50 runs).

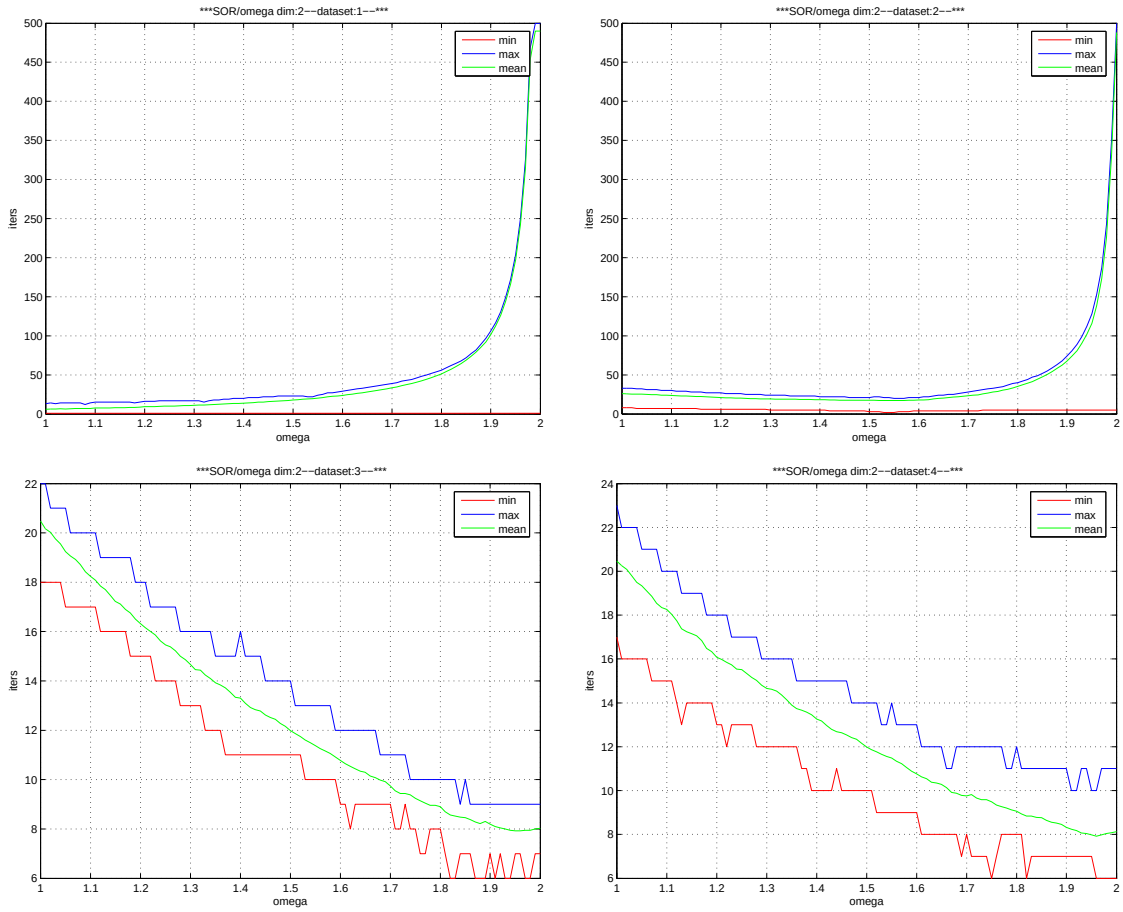


FIGURE 76 SOR relaxation parameter on data sets 1-4.

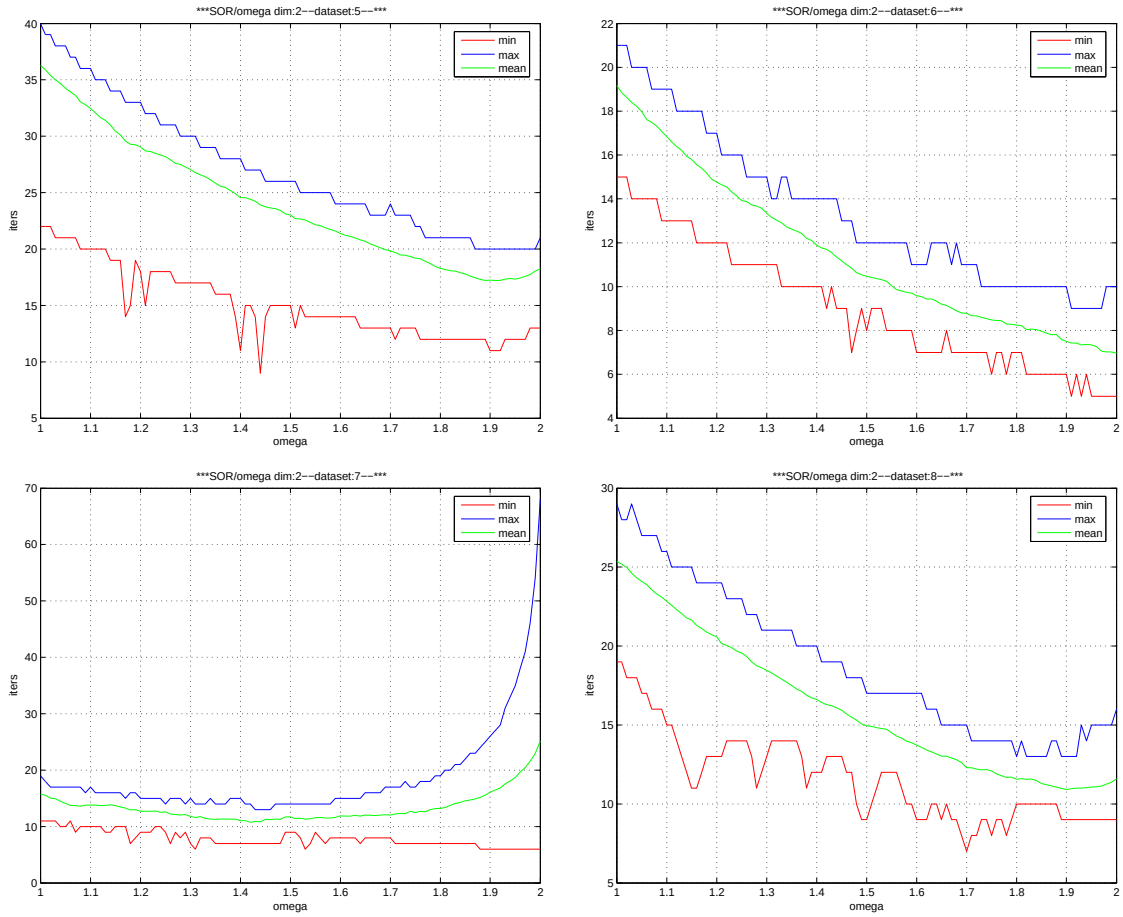


FIGURE 77 SOR relaxation parameter on data sets 5-8.

APPENDIX 3 ASSOR RELAXATION PARAMETER VALUE

The following figures depict the maximum, minimum and mean effect of relaxation parameter ω to the number of ASSOR iterations on the synthetic test data sets (the curves are averages over 50 runs).

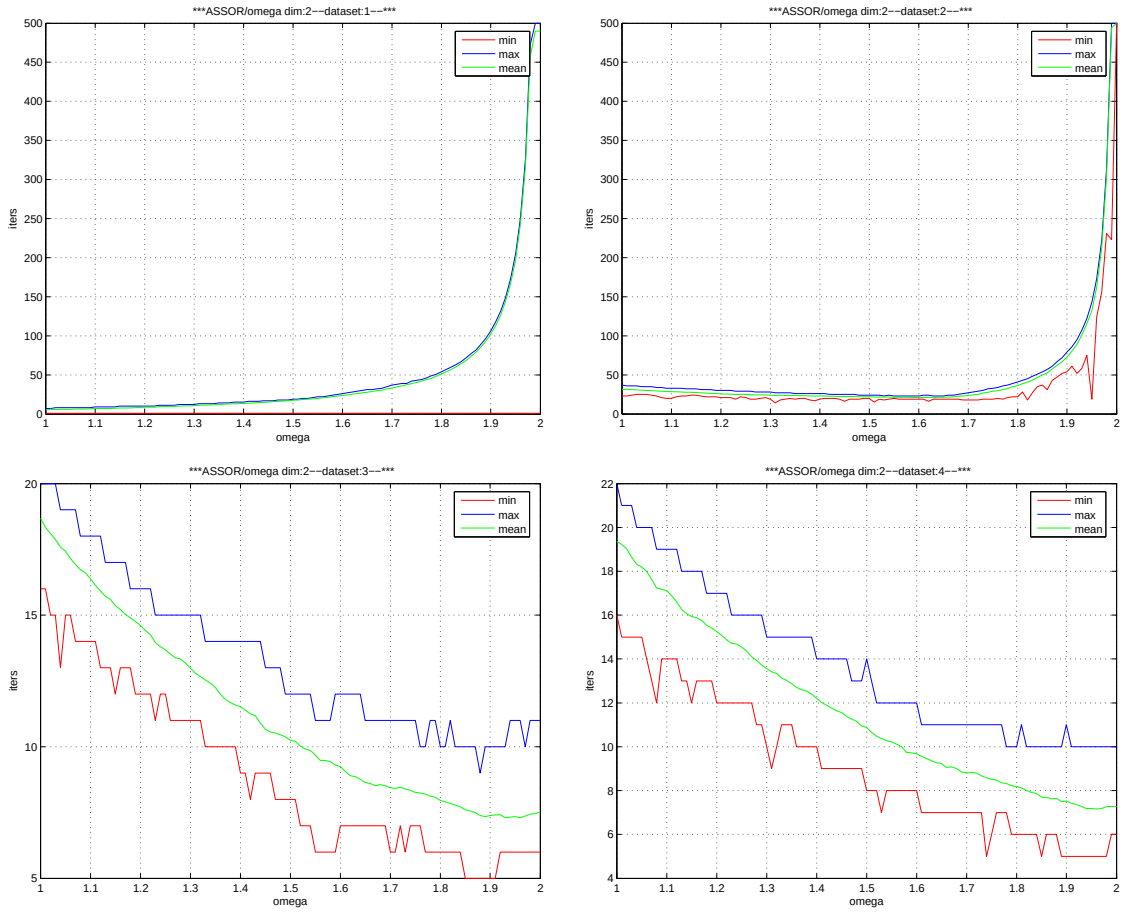


FIGURE 78 ASSOR relaxation parameter on data sets 1-4.

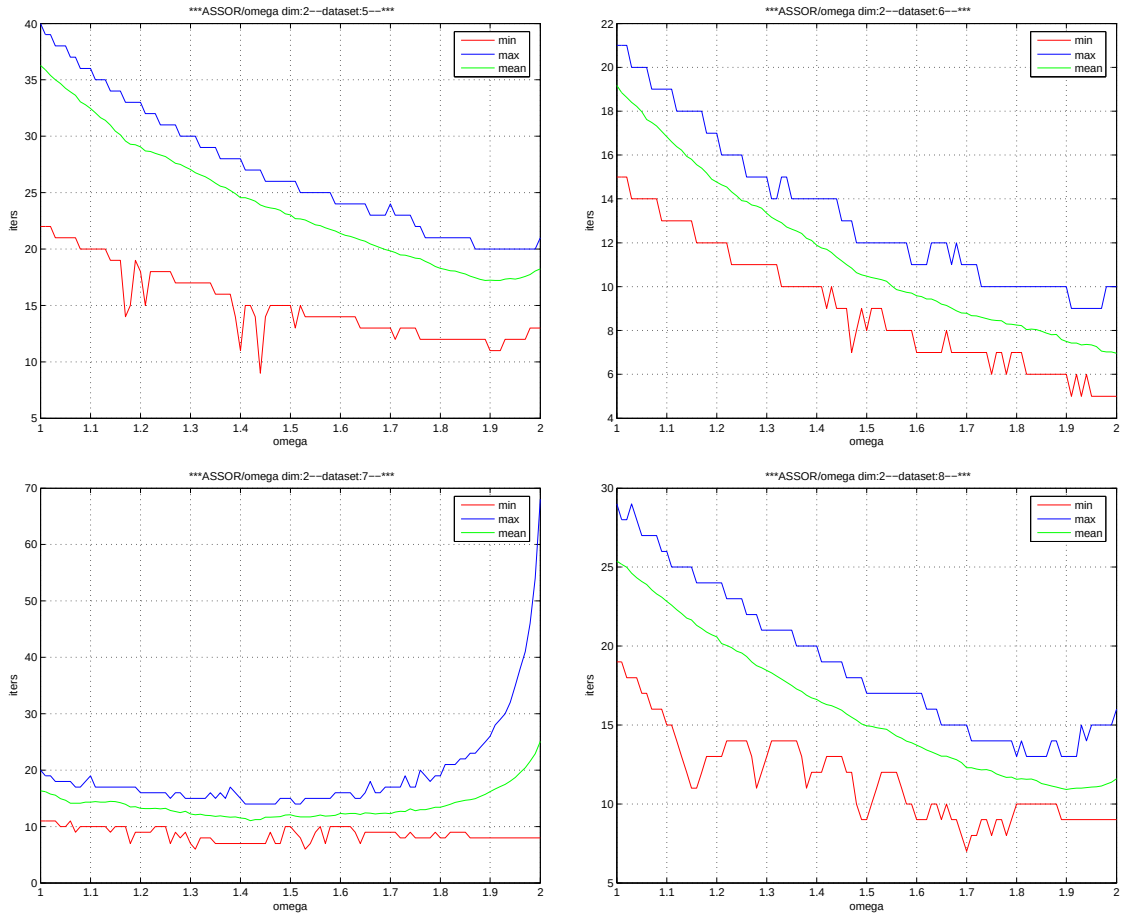


FIGURE 79 ASSOR relaxation parameter on data sets 5-8.

APPENDIX 4 PAPER INDUSTRY PROCESS DATA - CLUSTER VISUALIZATION

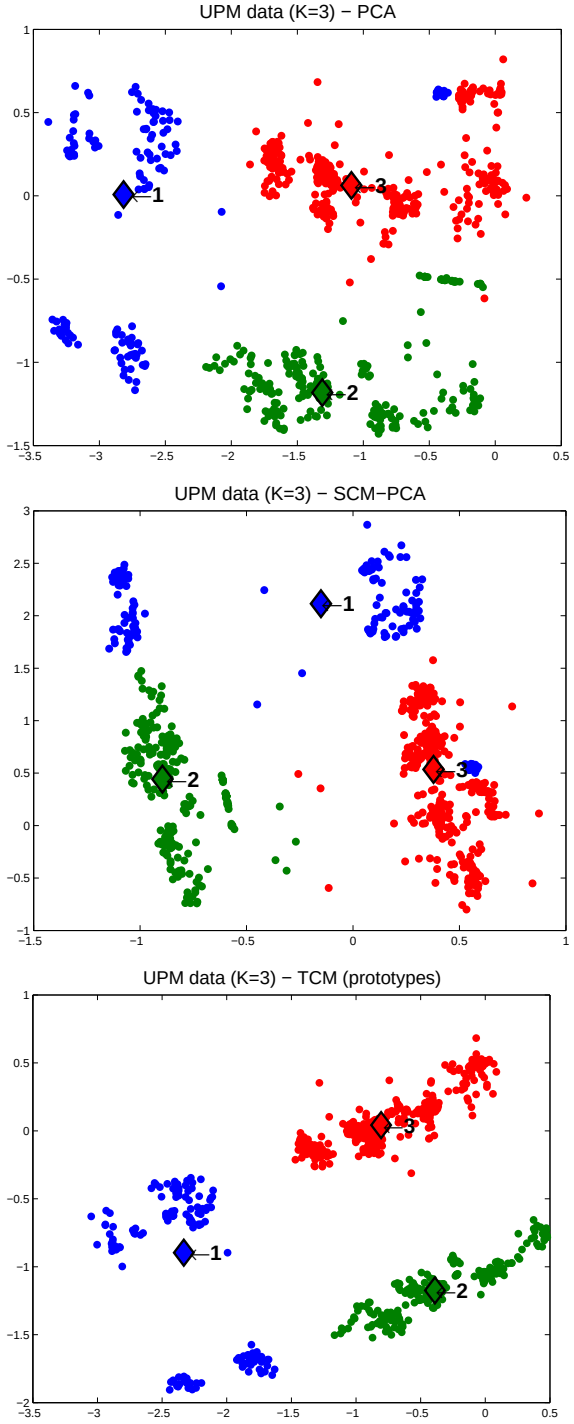


FIGURE 80 The classical, SCM, and TCM based principal component projections for 'paper data' in case $K = 3$.

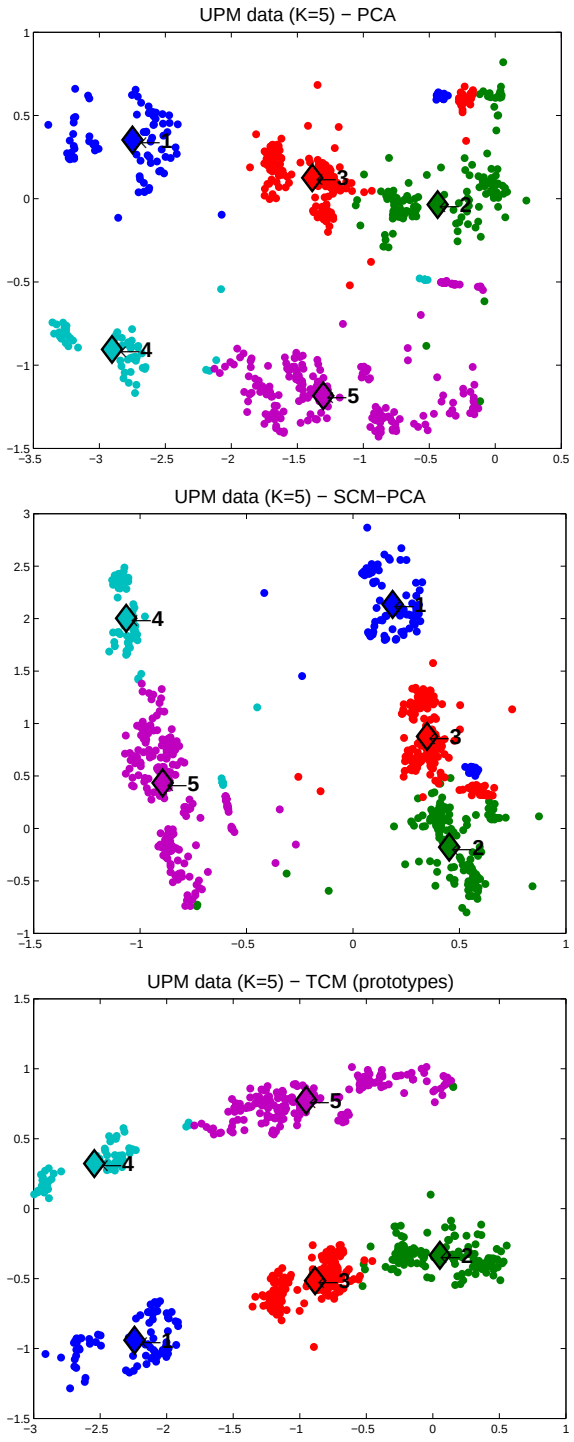


FIGURE 81 The classical, SCM, and TCM based principal component projections for 'paper data' in case $K = 5$.

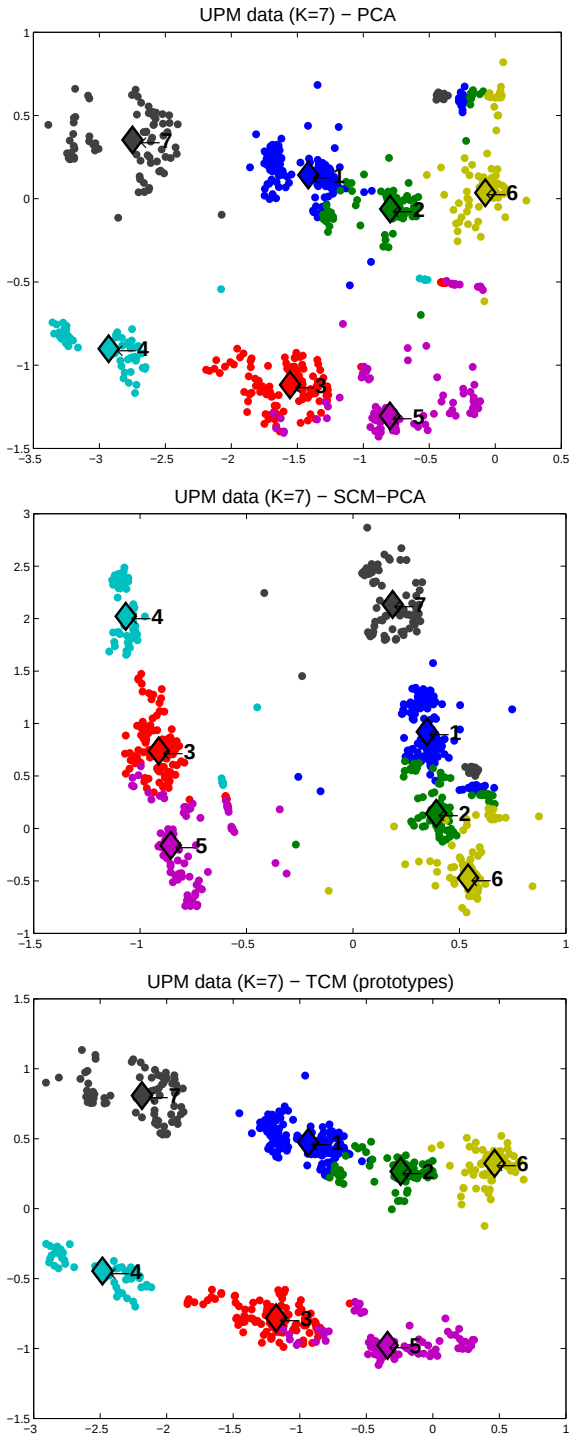


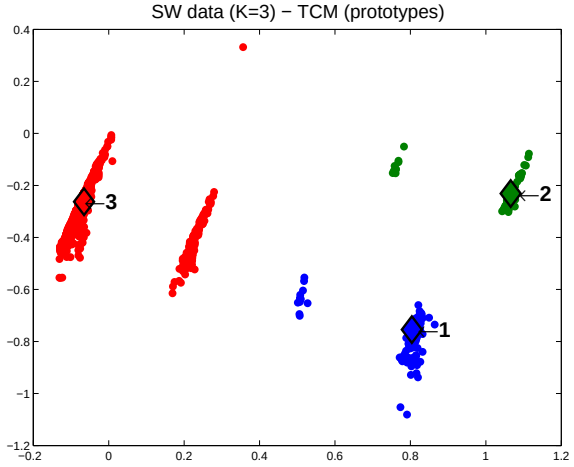
FIGURE 82 The classical, SCM, and TCM based principal component projections for 'paper data' in case $K = 7$.

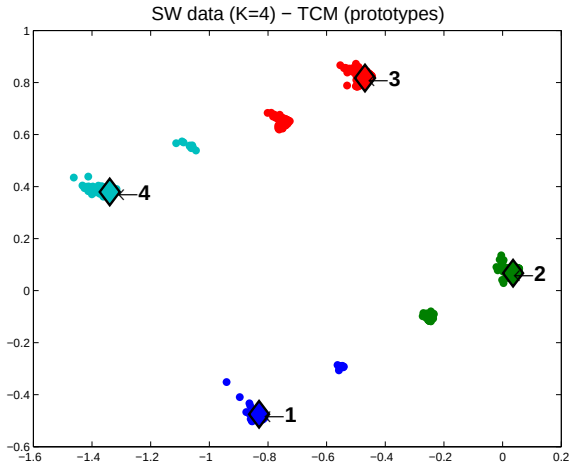
APPENDIX 5 ISBSG SOFTWARE PROJECT DATA - FIELD DESCRIPTIONS

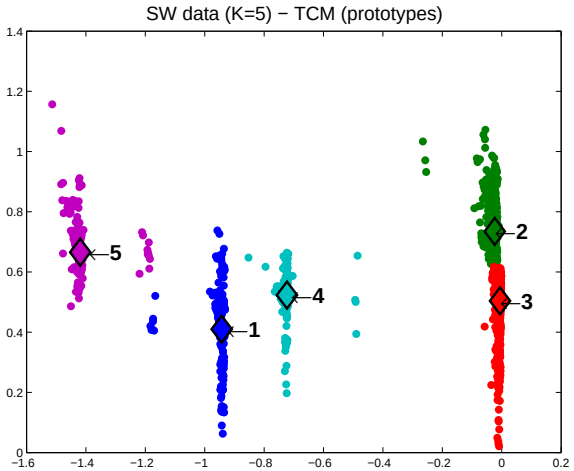
TABLE 24 Used software project data fields. See more detailed descriptions in [202].

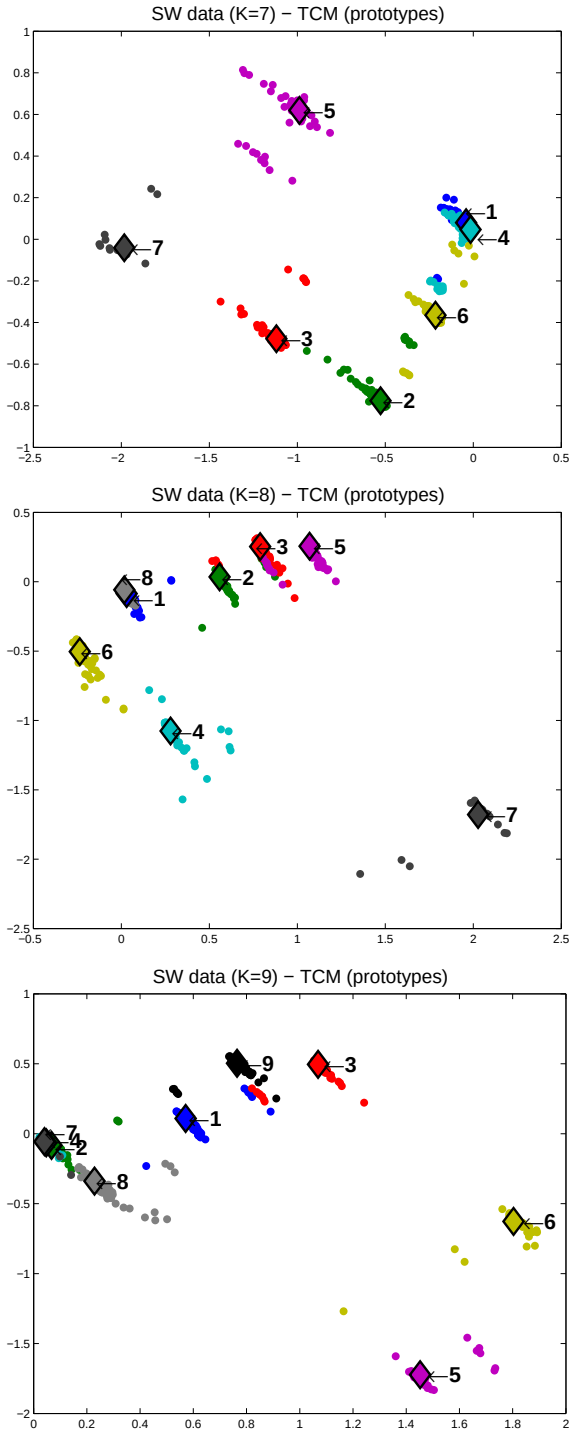
Index	Label
1	Functional Size
2	Adjusted Function Points
3	Value Adjustment Factor
4	Summary Work Effort
5	Normalized Work Effort
6	Reported PDR (afp)
7	Project PDR (ufp)
8	Normalized PDR (afp)
9	Normalized PDR (ufp)
10	Project Elapsed Time
11	Project Inactive Time
12	Effort Plan
13	Effort Specify
14	Effort Design
15	Effort Build
16	Effort Test
17	Effort Implement
18	Effort unphased
19	Minor defects
20	Major defects
21	Extreme defects
22	Total Defects Delivered
23	User Base - Business Units
24	User Base - Locations
25	User Base - Concurrent Users
26	Resource Level
27	Max Team Size
28	Average Team Size
29	Input count
30	Output count
31	Enquiry count
32	File count
33	Interface count
34	Added count
35	Changed count
36	Deleted count

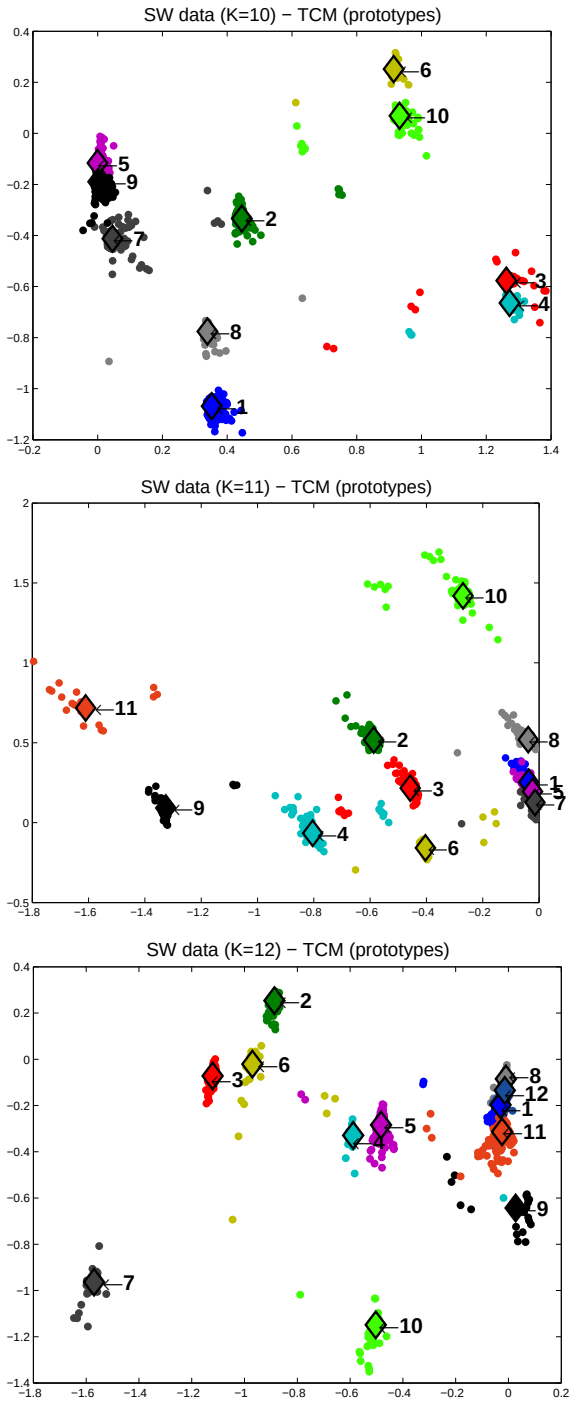
APPENDIX 6 SOFTWARE PROJECT DATA - CLUSTER VISUALIZATION

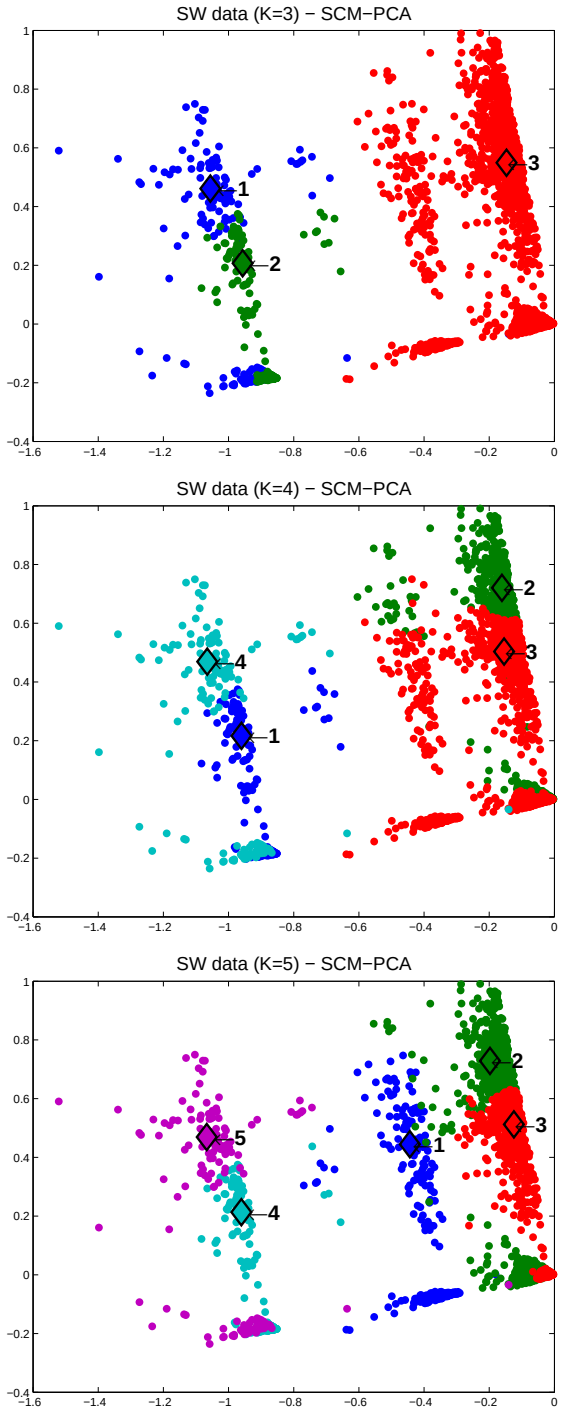


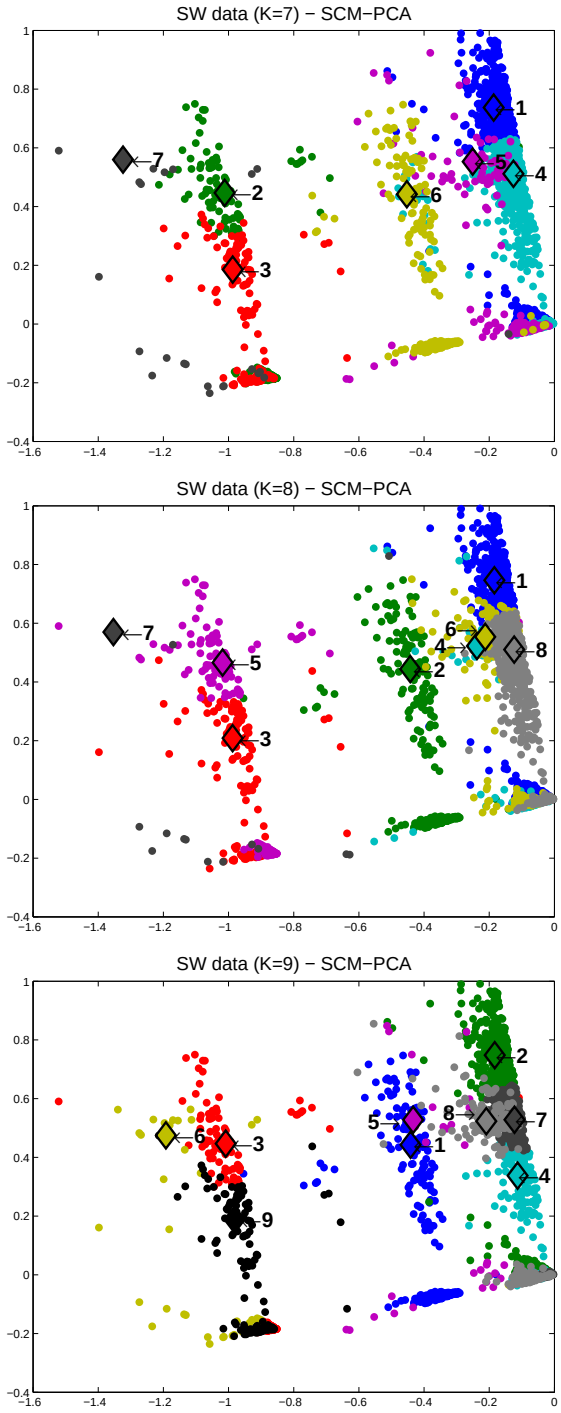


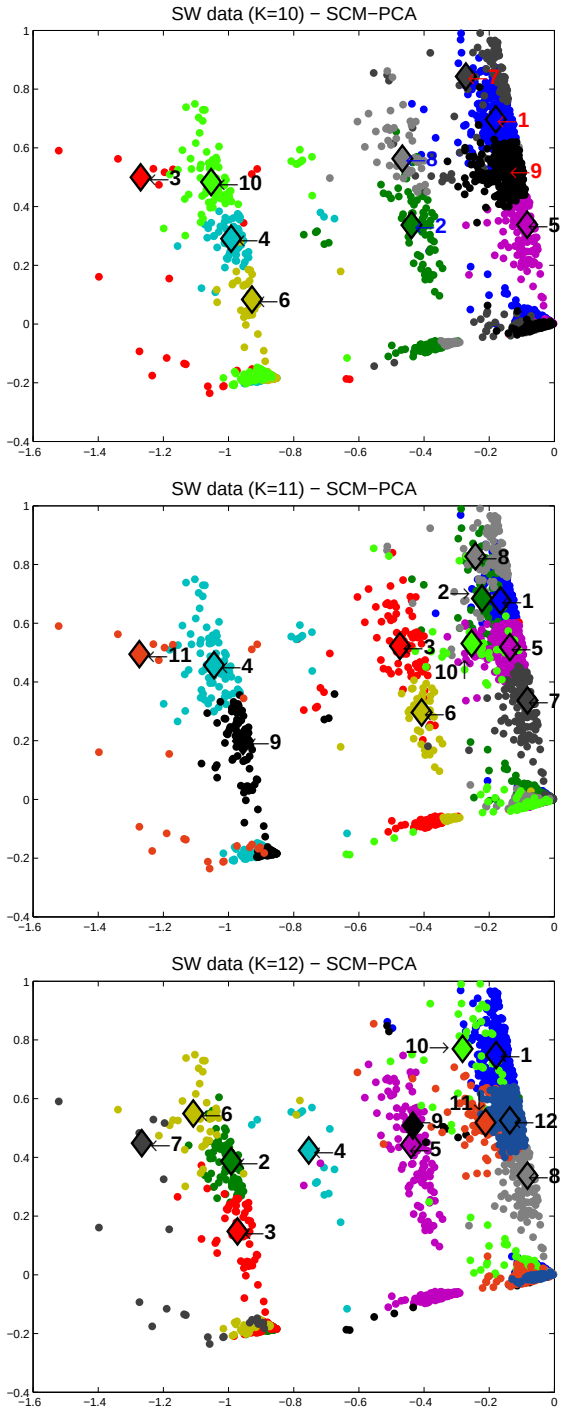


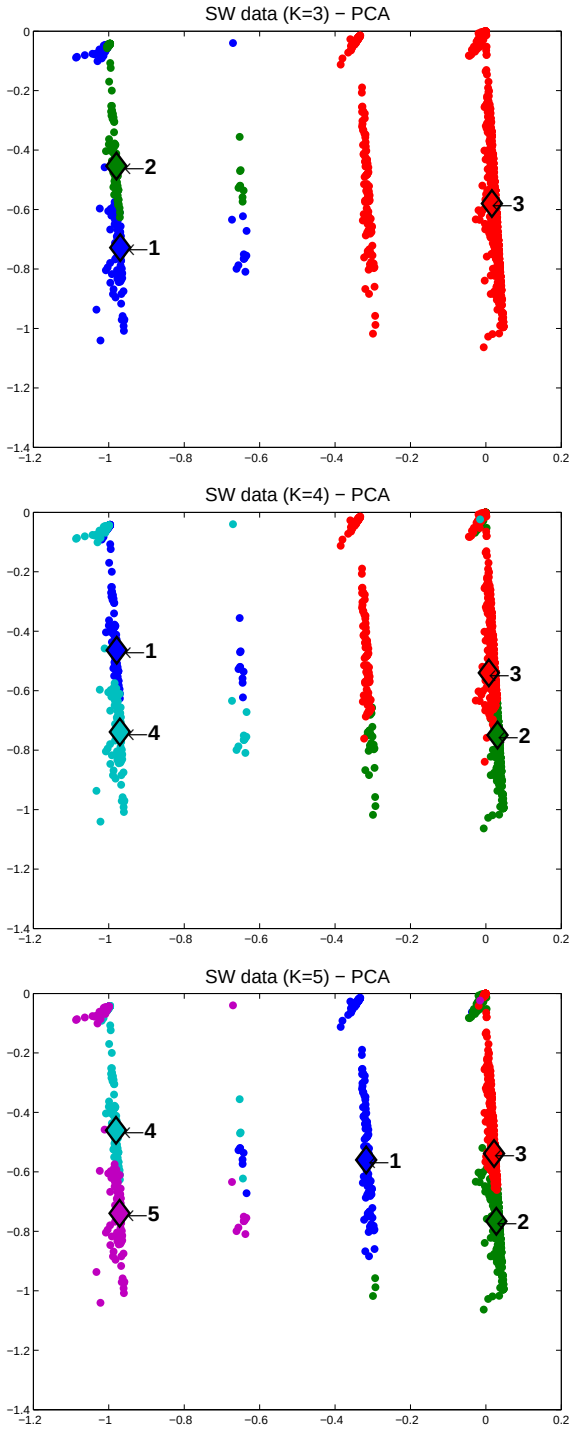

SW data (K=7) − TCM (prototypes)
SW data (K=8) − TCM (prototypes)
SW data (K=9) − TCM (prototypes)

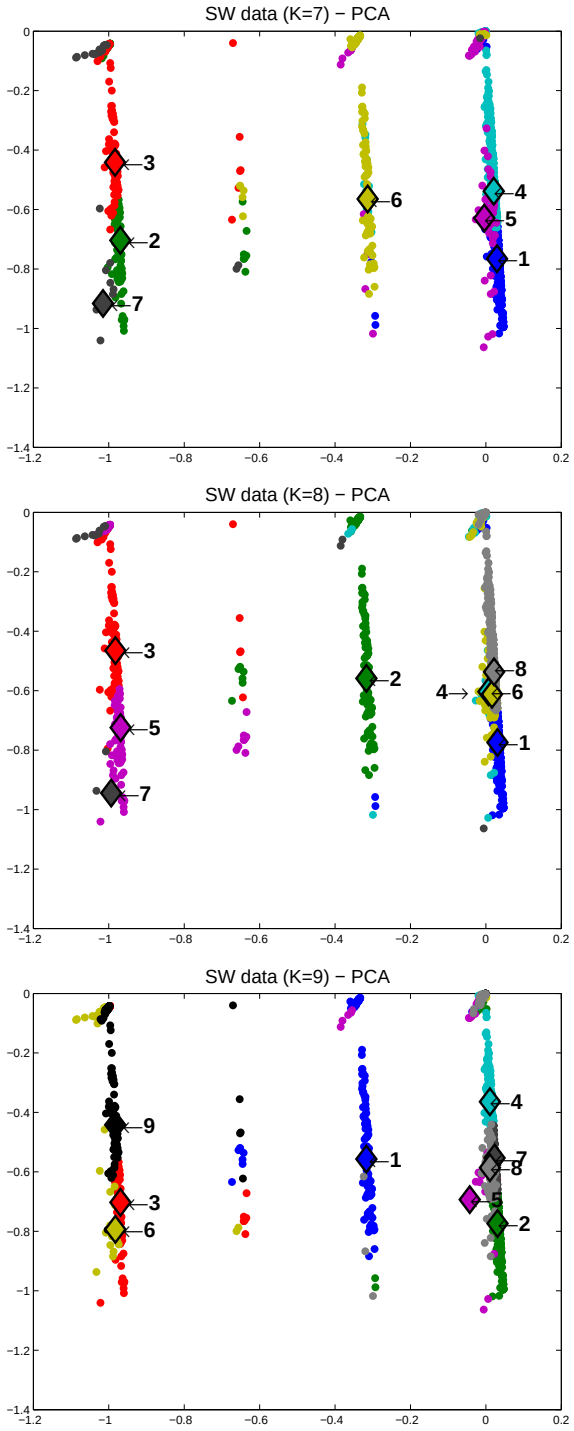

SW data (K=10) − TCM (prototypes)
6
10
5
9
2
7
3
4
8
1
SW data (K=11) − TCM (prototypes)
10
11
2
8
3
1
5
7
9
4
6
SW data (K=12) − TCM (prototypes)
2
6
3
8
12
1
5
11
9
7
10

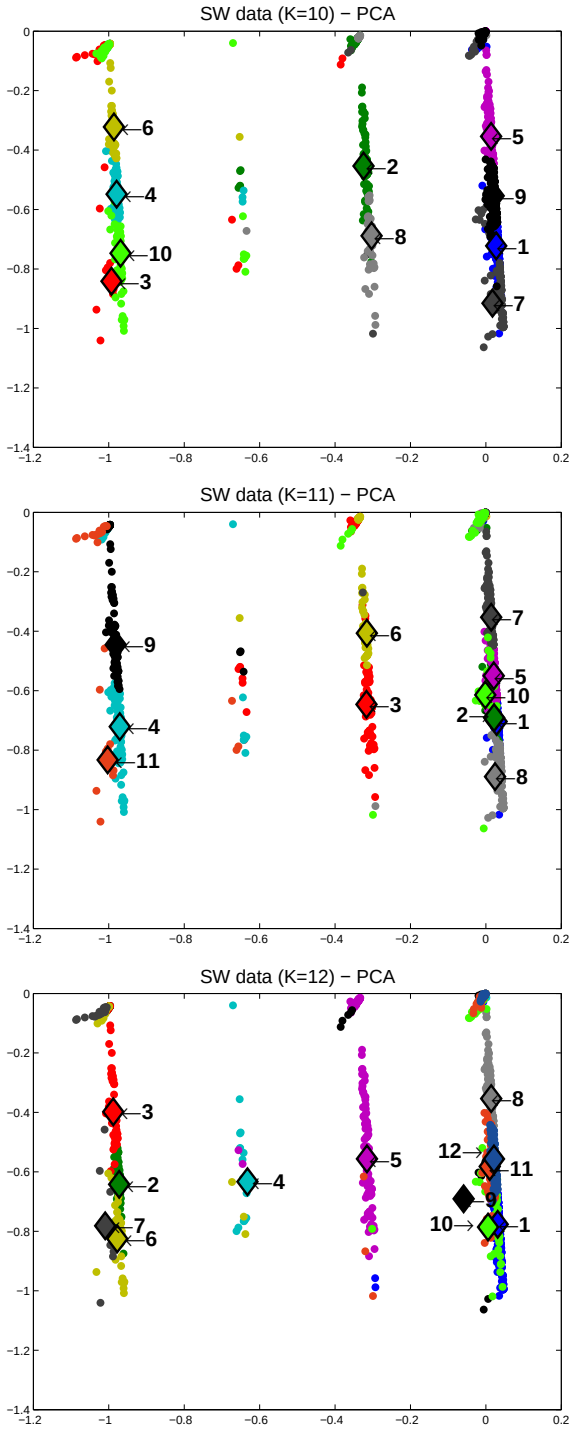

SW data (K=3) − SCM−PCA
1
2
3
SW data (K=4) − SCM−PCA
4
1
2
3
SW data (K=5) − SCM−PCA
5
4
1
2
3


SW data (K=7) − SCM−PCA
SW data (K=8) − SCM−PCA
SW data (K=9) − SCM−PCA


SW data (K=10) − SCM−PCA
SW data (K=11) − SCM−PCA
SW data (K=12) − SCM−PCA


SW data (K=3) − PCA
2
1
3
SW data (K=4) − PCA
1
4
3
2
SW data (K=5) − PCA
4
5
1
3
2


SW data (K=7) − PCA
3
2
7
6
4
5
1
SW data (K=8) − PCA
3
5
7
2
8
4
6
1
SW data (K=9) − PCA
9
3
6
1
4
7
8
5
2


SW data (K=10) − PCA
6
4
10
3
2
8
5
9
1
7
SW data (K=11) − PCA
9
4
11
6
3
7
5
10
2
1
8
SW data (K=12) − PCA
3
2
7
6
4
5
8
12
11
10
1

YHTEENVETO (FINNISH SUMMARY)

Erilaisten informaatiojärjestelmien kehittymisen ja yleistymisen seurauksena digitaalisesti tallennetun informaation määrä on kasvanut viime vuosina kiihtyvällä vauhdilla. Tämän seurauksena näiden tietomäärien hyödyntäminen on tullut yhä vaikeammaksi, sillä oleellinen informaatio saattaa monesti hukkua merkityksettömän tiedon, kohinan ja virheiden joukkoon. Tiedonlouhinnalla tarkoitetaan yleisesti ottaen uuden ja odottamattoman tietämyksen etsimistä suurista tietomassoista. Tavoitteena on esittää digitaalisen datan sisältämä informaatio käyttäjälle mahdollisimman ymmärrettävässä muodossa ja siten lisätä hänen tietämystään. Tässä tutkimustyössä kehitetään tilastollisesti luotettavia ja laskennallisesti tehokkaita ryhmittelymenetelmiä tiedonlouhintaan (engl. *data mining, DM*) ja tietämyksen etsimiseen (engl. *knowledge discovery in databases, KDD*) suurista tietomassoista. Kehitettyjen menetelmien avulla voidaan suuriakin määriä virheellistä ja puutteellista dataa jalostaa ja yksinkertaistaa käyttäjän kannalta ymmärrettävämpään muotoon.

Tämän työn päähuomio kohdistuu nk. prototyyppeihin perustuviin ryhmittelymenetelmiin, jotka voidaan toteuttaa iteratiivisia uudelleensijoittelualgoritmeja käyttäen. Prototyypillä tarkoitetaan tässä yhteydessä jonkin havaintoryhmän edustavinta jäsentä tai arvoa. Työssä käydään läpi prototyyppipohjaisiin ryhmittelymenetelmiin perustuvan dataryhmittelyn elementit ja kuvataan perusalgoritmit. Tutkimuksen päätuloksena esitellään datassa esiintyviä virheitä ja puutteita kestävä sekä loppukäyttäjän kannalta lähes automaattinen ryhmittelymenetelmä. Menetelmä koostuu useista eri osista, kuten alustus, prototyypin laskenta ja puuttuvan tiedon käsittely, jotka kukin on erikseen kehitetty ja testattu. Tilastollisten luotettavien tunnuslukujen matemaattisia ominaisuuksia on perusteellisesti tarkasteltu epäsileän optimoinnin näkökulmasta. Lisäksi työssä esitellään uusi mittari datajoukossa piilevien ryhmien lukumäärän arvioimiseen. Työssä on myös jatkokehitetty ja sovellettu menetelmiä, kuten esimerkiksi luotettavaa ja laskennallisesti tehokasta pääkomponenttianalyysia, datasta löytyvien ryhmien visuaaliseen havainnollistamiseen.

Työssä esitellään myös alkuperäisestä nk. KDD-prosessista tarkennettu KM (”*Knowledge Mining*”) -prosessimalli. Tämän tavoitteena on selventää ja täsmentää kehitettyjen menetelmien käytettävyyttä loppukäyttäjän kannalta. Uudessa mallissa painotetaan sovellusalueen analysoinnin tärkeyttä ja tiedonlouhinnan automaattista luonnetta loppukäyttäjän näkökulmasta. Prosessimallin merkitystä ja tärkeyttä on perusteltu esittelemällä kokoelma olemassa olevia tietämyksen lähteitä ja käytännön sovelluksia.

Menetelmien laskennalliset ja tilastolliset ominaisuudet kuten robustisuus, skaalautuvuus sekä laskennallinen ja tilastollinen tehokkuus, on testattu ja havainnollistettu useiden numeeristen kokeiden avulla. Numeeristen testien tueksi annetaan myös joitakin analyyttisia tuloksia ja käytännön esimerkkejä.